UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Rok Alidžanović

UVAJANJE AGILNE METODE SCRUM V RAZVOJ SPLETNEGA PORTALA ZA ZDRAVO PREHRANO

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Ljubljana, 2014

UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Rok Alidžanović

UVAJANJE AGILNE METODE SCRUM V RAZVOJ SPLETNEGA PORTALA ZA ZDRAVO PREHRANO

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: izr. prof. dr. Viljan Mahnič

Ljubljana, 2014

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Proučite agilno metodo Scrum in opišite postopek njenega uvajanja v razvoj spletnega portala za zdravo prehrano. Pri tem najprej predstavite namen portala in orodja, uporabljena za njegovo realizacijo, nato pa podrobno opišite priprave na uvedbo metode Scrum in potek projekta v prvih sedmih iteracijah. Pri tem posvetite posebno pozornost problematiki načrtovanja posameznih iteracij in merjenju napredka s pomočjo metrik, ki jih predlaga Scrum. Na koncu analizirajte pridobljene izkušnje in predlagajte, kako bi lahko uvajanje Scruma še izboljšali.

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Podpisani Rok Alidžanović, z vpisno številko **63960002**, sem avtor diplomskega dela z naslovom:

Uvajanje agilne metode Scrum v razvoj spletnega portala za zdravo prehrano

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom izr. prof. dr. Viljana Mahniča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 28. oktobra 2014 Podpis avtorja:

Zahvaljujem se mentorju izr. prof. dr. Viljanu Mahniču za njegove nasvete, popravke, spodbudo in ker me je bil pripravljen v tako kratkem času voditi do izdelave diplomskega dela. Zahvaljujem se tudi sodelavcem na projektu: Teji, Petru, Boštjanu in ostalim sodelavcem v podjetju. Svojim domačim bi se rad zahvalil za podporo v času študija. Še posebej pa bi se rad zahvalil svoji partnerki Gei, ki mi stala ob strani pri zaključevanju študija.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	Slovensko
CMM	Capability Maturity Model	zmožnostno zrelostni model
CMS	Content Management System	sistem za upravljanje vsebin
CSS	cascading style sheet	prekrivni slogi
CSV	comma separated values	z vejicami ločene vredosti v datoteki
FDD	Feature Driven Development	funkcionalno usmerjen razvoj
IDE	Integrated Development Environment	integrirana razvojna okolja
ISO	International Organization for Standardization	mednarodna organizacija za standardizacijo
MVC	Model–view–controller	model-pogled-kontroler
PHP	PHP Hypertext Preprocessor, izvirno pa Personal Home Page Tools	orodja za osebno spletno stran
PSD	Photoshop Document	dokument v grafičnem urejevalniku Photoshop
ROI	Return On Investment	donosnost naložbe
RSS	Rich Site Summary	zgoščeni povzetek strani
WIP	Work In Progress	delo v napredku
XHTML	extensible hipertext markup language	HTML, zapisan v sintaksi jezika XML
XML	extensible markup language	razširljivi označevalni jezik
XP	Extreme Programming	ekstremno programiranje

Povzetek

Diplomsko delo obravnava uvajanje agilne metode razvoja Scrum v razvoj spletnega portala za zdravo prehrano. Začetno poglavje je namenjeno seznanitvi z agilnimi metodami, ki v zadnjih letih postajajo vse bolj razširjene in uveljavljene pri razvoju programske opreme. Bolj podrobno je opisna agilna metoda Scrum.

V naslednjem poglavju se diplomsko delo osredotoči na predstavitev konkretnega projekta prenove spletnega portala, kjer je najprej predstavljeno poslanstvo in funkcija portala. Sledi opis nosilne platforme portala Magento in nekaterih orodij in storitev, ki so bile uporabljene pri razvoju: sistem za nadzor verzij Git in storitev Bitbucket.

Osrednje poglavje diplomskega dela je namenjeno postopku uvajanja agilne metode Scrum v razvoj portala. V tem poglavju je opisan postopek vzpostavitve začetnega seznama zahtev, načrtovanje posameznih iteracij, merjenje napredka s pomočjo metrik, ki jih predlaga Scrum in načrtovanje rokov in izdaj.

Zaključni del je namenjen analizi pridobljenih izkušnje in predlogom, kako bi lahko uvajanje Scruma še izboljšali.

Ključne besede: agilne metode, Scrum, Magento, Git, Bitbucket, JIRA, seznam zahtev, Planning Poker, načrtovanje iteracij, Sprint, merjenje napredka

Abstract

The thesis deals with the introduction of Scrum agile development method in the development of a web portal for a healthy diet.

The initial chapter is an introduction to the agile methods of software development which are becoming increasingly widespread and well-established during recent years. More specifically described is agile methode: Scrum.

In the next chapter, the thesis focuses on a renovation project of a web portal where it firstly presents the mission and function of the portal. Following is a description of the supporting platform Magento and some tools and services that were used during development of the portal: version control system Git and Bitbucket service.

The central section of the thesis is aimed at the process of introducing Scrum agile method in the development of the portal. This section describes the process of setting up an initial Product Backlog, the planning of iterations (Sprints), measuring progress using metrics proposed by Scrum and planning deadlines and releases.

The final part is devoted to the analysis of lessons learned and proposals of how to introduce Scrum even better.

Keywords: agile methods, Scrum, Magento, Git, Bitbucket, JIRA, Product Backlog, Planning Poker, planning of iterations, Sprint, measuring of progress

Poglavje 1 Uvod

Diplomska naloga obravnava postopek uvajanja agilne metode Scrum v razvoj spletnega portala za zdravo prehrano v podjetju, ki se ukvarja s prodajo prehranskih izdelkov ekološkega porekla.

Ker obstoječi spletni portal, ki podjetju služi že več kot pet let, postaja s svojim videzom in funkcionalnostjo zastarela, obenem pa spletne tehnologije neprestano napredujejo, je bila želja po prenovi prisotna že dlje časa. Obenem se podjetje želi plasirati na tujih trgih in distributerjem v tujini ponuditi različico portala, ki bo primerna za lokalizacijo v njihovem okolju. Tako je začel nastajati projekt prenove obstoječega spletnega portala.

Ker je pri projektu kazalo na to, da nabor zahtev pri prenovi ne bo popolnoma jasen že na samem začetku projekta in da bi se tekom projekta zahteve lahko spreminjale in dodajale, smo začeli razmišljati o možnosti uporabe agilnih metod za vodenje projekta. Avtor diplomskega dela sem v času formiranja projekta obiskoval predavanja pri predmetu Tehnologija programske opreme, kjer smo se seznanili z agilnimi metodami, v praktičnem primeru na vajah pa še prav posebej z metodo Scrum [34], ki se mi je s svojimi pristopi zdela zelo primerna za uporabo tudi na projektu prenove spletnega portala. V prid izboru metode Scrum je govorilo tudi dejstvo, da je bil razvoj obstoječega portala bolj ali manj domena enega razvijalca, za projekt prenove pa se je obetala razširitev ekipe, v kateri so se že dale prepoznati vloge metodologije Scrum.

Cilj projekta je torej izdelava novega spletnega portala s privlačnim in funkcionalnim grafičnim vmesnikom, ki bo primerna za uporabo na različnih sodobnih napravah, ki dostopajo do svetovnega spleta: poleg osebnih računalnikov tudi pametni telefoni televizije, tablice, itn. Obenem pa vidim cilj tudi v vpeljavi nove metode razvoja informacijskih rešitev, ki bo podjetju služila tudi v bodoče.

V drugem poglavju bom najprej opisal sodobne metode agilnega razvoja programske opreme, njihov nastanek, raznolikost in uporabo. Posebej bom opisal metodo Scrum od njenega nastanka do uporabe v tehnologiji programske opreme. V tretjem poglavju bom podrobneje opisal projekt prenove spletnega portala skozi njegovo poslanstvo in funkcijo. Posvetil sem bom tudi odprtokodni platformi Magento, na kateri deluje obstoječ in bo delovala tudi nov

portal. V četrtem poglavju pa sledi konkreten prikaz uvajanja metode Scrum v projekt prenove spletnega portala. Opisal bom, kako se je formirala ekipa, kako smo izbrali orodje za vodenje projekta, kako projekt v dosedanjih sedmih iteracijah potekal in kako smo pri tem reševali različne izzive in težave. Diplomsko delo bom zaključil s poglavjem sklepnih ugotovitev, kjer se bom ozrl nazaj in povzel bistvena dognanja, do katerih smo prišli tekom projekta, in pogledal naprej, kako bi ta dognanja uporabili v prihodnje.

Poglavje 2 Agilne metode za razvoj programske opreme

V tem poglavju bom opisal osnovne smernice agilnih metod razvoja in se osredotočil na eno od pogosto uporabljenih metod pri agilnem razvoju: Scrum.

2.1 Glavne značilnosti

Agilne metode razvoja programske opreme so nastale kot alternativa tradicionalnim metodam projektnega vodenja. Značilnosti agilnih metod, ki jih podaja Manifest agilnosti [27] v kontekstu agilnega razvoja programske opreme, so naslednje:

- posamezniki in interakcije: pri agilnem razvoju so pomembne vrednote posameznika, kot so samoorganiziranost in motivacija, kot tudi interakcije v smislu intenzivnega sodelovanja z uporabniki in med samimi razvijalci (npr. programiranje v parih);
- delujoča programska oprema, ki zagotavlja zadovoljstvo strank v nasprotju s kupom podrobne dokumentacije;
- sodelovanje z uporabniki: ker zahteve ne morejo biti zbrane v popolnosti že na začetku razvojnega cikla, je pomembna nepretrgana vpletenost uporabnikov v proces razvoja;
- odzivnost na spremembe: agilni razvoj se osredotoča na hitro odzivnost na spremembe in nepretrgan razvoj [12].

2.2 Zgodovina uporabe

Februarja leta 2001 se je v letovišče Snowbird v ameriški zvezni državi Utah zbrala skupina 17 razvijalcev programske opreme z namenom, da bi razpravljali o *lahkih* metodah razvoja. Tako je nastal Manifest agilnosti [27]. Nekateri od avtorjev manifesta so ustanovili neprofitno zvezo Agile Alliance, ki spodbuja razvoj programske opreme po principih in vrednotah manifesta.

Kasneje je Ken Schwaber s somišljeniki ustanovil zvezo Scrum Alliance, ki je ponudila sistem certificiranja *Certified Scrum Master*. Potem, ko je zapustil zvezo Scrum Alliance, je

leta 2009 ustanovil organizacijo Scrum.org [33], ki nudi izobraževanja in pisane vodiče po agilni metodi Scrum, kateri se bom posvetil kasneje v poglavju.

Leta 2005 je skupina pod vodstvom Alistaira Cockburna in Jima Highsmitha napisala dodaten dokument o principih projektnega vodenja po imenu Declaration of Interdependence [2] z namenom, da bi začrtala smernice projektnega vodenja po principih agilnih metod.

Leta 2009 pa je gibanje pod vodstvom Roberta Martina spisalo dokument z razširitvijo principov razvoja programske opreme po imenu Software Craftsmanship Manifesto [28] z namenom obrazložitve vodenja agilnega razvoja s profesionalnim ravnanjem in mojstrstvom.

2.2.1 Bistvene razlike med tradicionalnimi in agilnimi metodami razvoja programske opreme

Pri tradicionalnih discipliniranih pristopih razvoja programske opreme izdelava poteka po točno določenem, vnaprej predpisanem postopku. Projekt se običajno razbije na zaporedje majhnih, obvladljivih korakov, pogosta je uporaba diagramskih tehnik, kot so npr. entitetno-relacijski diagrami, diagrami toka podatkov in objektno orientirani diagrami. Načrtovanje je natačno in poteka vnaprej, rezultat tega pa je pogosto obsežna dokumentacija in posledično slaba odzivnost na spremembe v zahtevah. Pogosto so uporabljeni tudi modeli za zagotavljanje kakovosti, kot so zmožnostno zrelostni model [14] (CMM - Capabality Maturity Model) in standard ISO 9001 [19] (ISO - International Organization for Standardization). Drugi problemi, ki so značilni za disciplinirane pristope, so:

- togost, ki predpisuje zaporedje korakov,
- dolg življenjski cikel, pri katerem naročnik šele na koncu dobi izdelek in
- še vedno veliko število neuspešnih projektov.

Agilni pristopi pa se za razliko od tradicionalnih prilagajajo spremembam v zahtevah uporabnika in opuščajo aktivnosti, ki niso nujno potrebne. Ker pa je agilen pristop razmeroma nov, zahteva podrobnejše ovrednotenje posameznih konceptov, čeprav precej študij kaže, da je uspešnejši od tradicionalnega.

Težko je zaključiti, kateri pristop je boljši, saj je vsak dober na svojem domačem terenu, v splošnem pa velja, da je potrebo iskati ravnovesje med obema [6].

2.2.2 Raznolikost agilnih metod

Agilni pristopi v sodobnem času obsegajo veliko različnih metod, od katerih jih bom naštel samo nekaj:

- Ekstremno programiranje (XP - eXtreme Programming): metodologija, ki zagovarja uporabo že znanih principov in praks v ektremni obliki: npr. pregledovanje kode se nadgradi v programiranje v parih, testiranje se nadgradi v vnaprejšnjo pripravo testov in sprotno testiranje, načrtovanje v sprotno preoblikovanje kode, ipd. [12].

- *Vitki* razvoj programske opreme (LSD - lean software development), čigar filozofija smatra vse, kar naročniku ne predstavlja dodane vrednosti kot odpadek (npr. nepotrebna programska koda in funkcionalnost, zamude v procesu razvoja, nejasne zahteve, ipd.) [22].

- Kanban je novejši pristop k razvoju programske opreme, ki kombinira agilne in vitke principe. Njegova osnovna ideja je, da omeji delo v stanju napredovanja (WIP - Work In Progress) s čimer zagotavlja trajnostni ritem razvoja [12].

- Funkcionalno usmerjen razvoj (FDD - feature driven development) je iterativen in inkrementalen razvoj programske opreme, ki naročniku zagotavlja pravočasno dostavo *otipljive* in delujoče programske opreme [17].

- Scrum, kateremu se bom posvetil v poglavju 2.3.

2.2.3 Uporaba agilnih metod v sodobnem času

Številne raziskave kažejo na to, da agilne metode in prakse v skupnosti razvijalcev programske opreme v zadnjih letih doživljajo široko sprejetje.

Januarja 2010 je Forrester [10] objavil rezultate svoje ankete Global Developer Technographics Survey, ki je razkrila, da 35% anketirancev uporablja agilni proces razvoja. V istem času je Gartner [8] napovedoval, da bodo agilne metode do leta 2012 uporabljene v 80% vseh projektov razvoja programske opreme.

Od leta 2006 se izvaja raziskava po imenu *The State of Agile* Survey [35] (slov. stanje agilnih metod), ki vključuje tisoče udeležencev iz skupnosti razvoja programske opreme. Raziskava spremlja trende dobrobiti, ki jih prinašajo agilne metode, kot npr. pridobljene izkušnje in zaželene prakse. Rezultati raziskave iz leta 2013, ki so bili izdani januarja 2014, zaključujejo:

- 52% anketirancev zagotavlja, da uporablja agilne metode pri vodenju večine svojih projektov,
- da 73% anketirancev zagotavlja, da jim agilni pristopi omogoča hitrejše dokončanje projektov,
- 92% anketirancev zagotavlja, da agilni pristopi izboljšujejo njihovo zmožnost prilagajanja prioritetam naročnikov,
- 87% anketirancev zagotavlja, da agilni pristopi izboljšujejo produktivnost njihovih razvojnih ekip.

Skupina Standish Group v svojem poročilu *CHAOS Report* ugotavlja, da je bilo leta 2011 uspešnih 14% tradicionalnih (waterfall) projektov in 42% agilni projektov [16].

Rezultati anket torej pričajo o tem, da popularnost in uspešnost agilnih metod naraščata.

2.2.4 Uporaba orodij pri vodenju projektov

Glede na zadnjo raziskavo State of Agile [36] so najpogostje uporabljena orodja pri vodenju projektov naslednja: Excel, Microsoft Project, VersionOne, JIRA, Microsoft TFS idr.

Zadovoljstvo uporabikov pa zanimivo narašča ravno pri orodjih, ki so bila razvita prav za agilne metode. To so: VersionOne (93% zadovoljnih uporabnikov), JIRA (87%), Vendor Y (ki ni hotel biti poimenovan, 85%), LeanKit (84%) in Target Process (83%).

2.3 Metoda Scrum

Scrum je metoda in ogrodje za razvoj in vzdrževanje kompleksnih produktov. Definicija metode Scrum pravi, da gre za ogrodje, znotraj katerega ljudje lahko naslavljajo kompleksne, prilagodljive probleme, medtem ko produktivno in kreativno dostavljajo produkte najvišje mogoče kvalitete. Metodologija Scrum je: *lahka* (angl. lightweight), preprosta za razumevanje, vendar težka, da bi jo popolnoma obvladali [9].

V kontekstu razvoja programske opreme razlika med definiranimi in empiričnimi procesi zahteva drugačen način vodenja, ker je proces razvoja programske opreme preveč zapleten in nepredvidljiv, da bi ga lahko natančno planirali vnaprej. Potreben je stalen vpogled, nadzor in prilagajanje, kar dosežemo z razvojnim procesom, ki je iterativen in inkrementalen [6]. Te lastnosti pa zagotavlja metodologija Scrum.

2.3.1 Nastanek

Metodo Scrum (pod tem imenom) sta razvila Ken Schwaber in Jeff Sutherland v zgodnjih devedesetih prejšnjega stoletja, čeprav njeni začetki segajo že v osemdeseta leta pejšnjega stoletja, ko sta avtorja Hirotaka Takeuchi in Ikujiro Nonaka metodo opisovala kot holističen, ragbi (angl rugby) pristop (pri ragbiju scrum pomeni ponoven začetek igre po manjši prekinitvi). Ta nov pristop k razvoju komercialnih produktov naj bi povečal hitrost in fleksibilnost na podlagi vzorčnih študij iz proizvodnih podjetij, kot so avtomobilska industrija in proizvodnja fotokopirnih strojev in tiskalnikov. V nasprotju s tradicionalnim zaporednim pristopom gre za fleksibilno in holistično strategijo, pri kateri ekipa deluje kot celota z namenom, da doseže zastavljen cilj [34].

2.3.2 Opis metode

Ogrodje metode Scrum sestavljajo ekipe in njihove povezane vloge, dogodki in pravila. Vsaka komponenta znotraj ogrodja služi določenem namenu in je bistvena za uporabo in uspeh metodologije. Pravila metode Scrum združujejo vloge, dogodke, izdelke in narekujejo njihovo medsebojno interakcijo [9].

2.3.2.1 Vloge v metodologiji Scrum

Ekipo metode Scrum, katere značilnost je samoorganizacija in navzkrižna funkcionalnost, sestavljajo naslednje vloge:

- Lastnik izdelka (angl. Product Owner), ki predstavlja vse, ki so zainteresirani za projekt in njegove rezultate. Lastnik izdelka skrbi za zagotavljanje sredstev in donosnost naložbe (ROI - return on investment). Prav tako je odgovoren za upravljanje seznama zahtev, kar vključuje: izdelavo začetnega seznama, določanje prioritet posameznih zahtev in planiranje predaje posameznih verzij izdelka v obratovanje.

- Razvojna skupina, ki odgovarja za razvoj novih funkcionalnosti znotraj vsake iteracije. Scrum znotraj razvojne skupine ne predpisuje drugih nazivov kot razvijalec ne glede na delo, ki ga oseba opravlja. Prav tako ne priznava nikakršne hierarhije med lani skupine, ne glede na njihovo funkcijo (npr. testiranje ali poslovna analitika). Člani skupine so kolektivno odgovorni za uspeh posameznih iteracij in projekta v celoti. Velikost razvojne ekipe mora biti dovolj majhna, da ostane prilagodljiva in dovolj velika, da zaključi zadostno količino opravil znotraj ene iteracije. Priporočilo govori o velikosti skupine od 3 do 9 članov.

- Skrbnik metodologije Scrum odgovarja za skladnost celotnega procesa z metodologijo. Deluje v vlogi vodje in služabnika ekipi, uči metodologijo vse, ki sodelujejo na projektu in pomaga maksimirati dodano vrednost, ki jo proizvede ekipa. Obenem skrbi, da se metodologija vklaplja v kulturo organizacije, a še vedno daje pričakovane rezultate.

- Opazovalci so ostali zainteresirani za projekt, ki se po metodologiji ne smejo neposredno vmešavati [3][6].

Samoorganizirane ekipe same izberejo, kako bodo najbolje opravile delo, namesto da bi bile diktirane od zunaj [9].

2.3.2.2 Izdelki metodologije Scrum

Izdelki predstavljajo delo ali vrednost, ki nudi transparentnost in možnost za pregled in prilagoditev. Izdelki, ki jih definira Scrum, so:

- Seznam zahtev (angl. Product Backlog), ki ni nikoli dokončen. Na začetku projekta predstavlja začetni nabor zahtev, med delom na projektu pa se dopolnjuje in postaja bolj podroben z namenom, da bi končni izdelek postal primeren, konkurenčen in uporaben. Obstaja ves čas, dokler je izdelek v uporabi. Seznam zahtev obsega vse funkcionalnosti, izboljšave in popravke, ki predstavljajo spremembe za izvedbo v bodočih izdajah. Zahteve so grupirane v iteracije (Sprint), iteracije pa v izdaje na podlagi prioritet. Zahteve, ki niso realizirane v eni iteraciji, se prenesejo v eno od naslednjih iteracij. Prioritete se med izvedbo lahko spreminjajo.

- Seznam iteracije (Sprint Backlog) je seznam nalog, ki morajo biti opravljene v eni iteraciji. Seznam iteracije obsega tudi načrt za dostavo dodane funkcionalnosti, ki je neposredno uporabna, in načrt za dosežek cilja iteracije. Obseg dela za vsako nalogo mora znašati 4 do 16 ur, zato je grobo definirane naloge potrebno razbiti na več manjših. Seznam iteracije lahko spreminja samo razvojna ekipa. Vsaka naloga v seznamu na določen dan vsebuje oceno dela, ki še ni bilo opravljeno. Na ta način razvojna ekipa lahko spremlja napredek tekom iteracije.

- Prirastek (angl. increment) ali dodana funkcionalnost, ki je neposredno uporabna. Prirastek obsega seštevek vseh nalog, ki so bile zaključene tekom ene iteracije in dodano vrednost vseh prejšnjih iteracij, ki vključuje tudi dokumentacijo v obliki navodil in vgrajenih datotek za pomoč. Novi prirastek mora ustrezati konceptu *Done* (slov. narejeno), kar pomeni v celoti stestirano, dobro strukturirano in dobro napisano

kodo. Prirastek mora biti uporaben ne glede na to, ali se lastnik produkta odloči, da bo izdan [6], [9].

2.3.2.3 Sestanki metode Scrum

Metoda Scrum predpisuje 4 tipe sestankov:

- *Daily Scrum* je vsakodnevni 15-minutni sestanek razvojne skupine, katerega namen je usklajevanje njenih aktivnosti in planiranje za naslednjih 24 ur. V praksi se izvrši tako, da vsak član ekipe odgovori na 3 vprašanja:
    - Kaj si delal od prejšnjega sestanka?
    - Kaj nameravaš delati do naslednjega sestanka?
    - S katerimi težavami se srečuješ pri delu?
- *Sprint planning meeting* je dogovor med lastnikom izdelka in razvojno skupino o zahtevah, ki bodo realizirane v naslednji iteraciji (sprintu). Traja 8 ur in je sestavljen iz dveh delov po 4 ure: najprej se izbere zahteve, potem pa se izdela začetna verzija seznama iteracije.
- *Sprint review meeting* je sestanek, kjer se vsem zainteresiranim za projekt predstavi rezultate na koncu vsake iteracije. Glede na to udeleženci sestanka sodelujejo pri tem, kaj bi bilo dobro zaključiti v naslednji iteraciji, da bi optimizirali dodano vrednost.
- Sprint retrospective meeting je sestanek, kjer sodelujejo samo člani delovne skupine. Vodi ga skrbnik metodologije in vključuje oceno preteklega dela in plan izboljšav za naslednjo iteracijo [6], [9].

2.3.2.4 Nadzor nad potekom dela

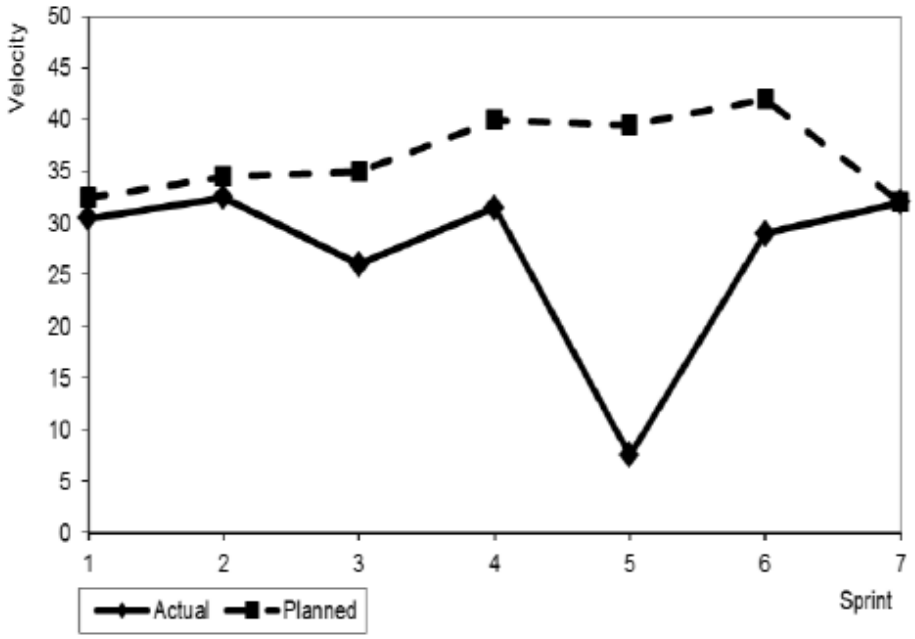
Za nadzor nad potekom dela se uporabljajo naslednje mere: hitrost (angl. Velocity) in količina preostalega dela, ki jo predstavljata diagram napredka izdaje (angl. Release burndown chart) in diagram napredka iteracije (angl. Sprint burndown chart) [7].

Velocity je vrednost, ki predstavlja količino zaključenega dela v vsaki iteraciji, izražena s seštevkom točk zaključenih uporabniških zgodb. Razvojna ekipa oceni planirano hitrost na začetku vsake iteracije in na podlagi te ocene v iteracijo za izvedbo uvrsti zadostno število uporabniških zgodb, katerih seštevek točk v ocenah zahtevnosti ustreza predvideni hitrosti. Dejanska hitrost se izračuna na koncu iteracije s seštevanjem točk uporabniških zgodb, ki so

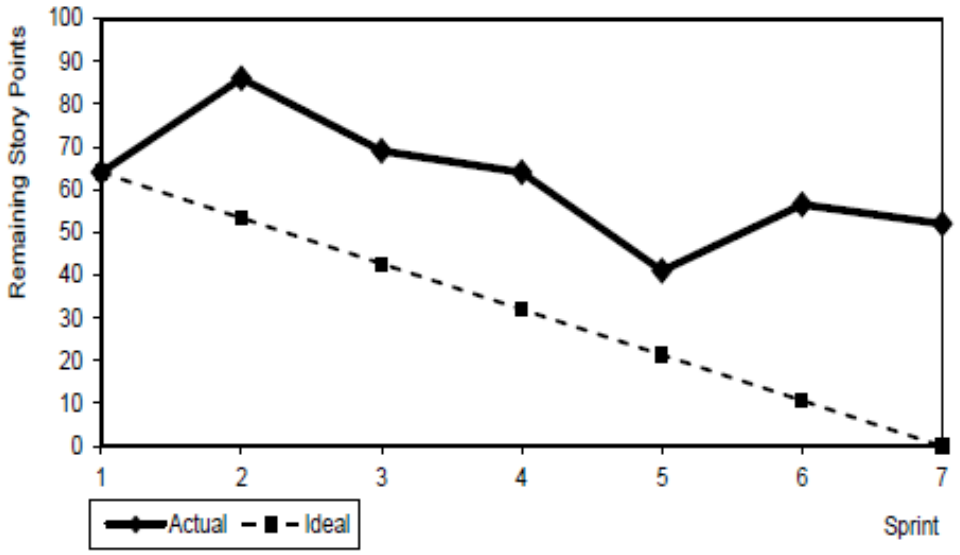
sprejete s strani lastnika izdelka. Spremljanje napredka projekta zahteva stalno primerjanje planirane in dejansko dosežene hitrosti. Primer ustreznega diagrama je prikazan na sliki 2.1.

Release burndown chart je diagram, ki prikazuje količino preostalega dela na začetku vsake iteracije z izrisom seštevka točk uporabniških zgodb, ki do takrat še niso bile zaključene. Tako pokaže korelacijo med preostalim delom in napredkom razvojne ekipe, ki zmanjšuje to delo. Slika 2.2. prikazuje primer tega diagrama na začetku 7. iteracije.

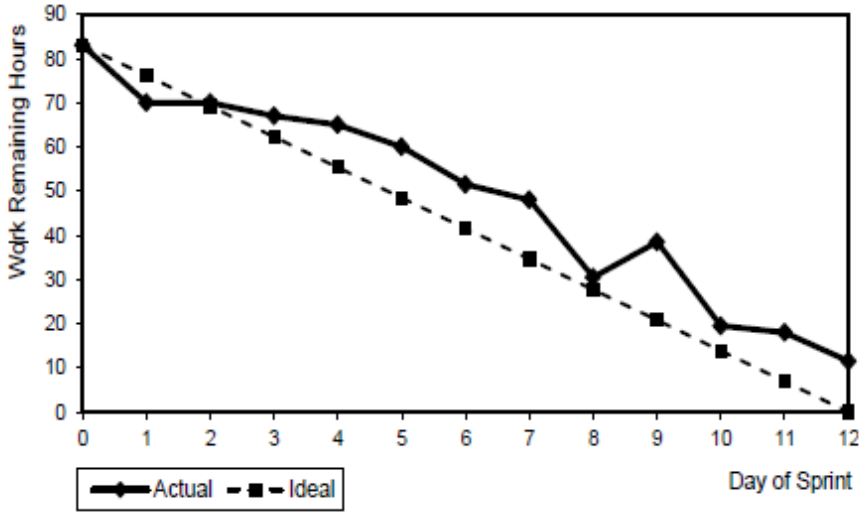
Sprint burndown chart je diagram, ki je podoben diagramu *Release burndown chart*, vendar namesto širše slike prikazuje preostanek dela, ki naj bi bilo zaključeno do konca iteracije. Horizontalna os prikazuje dneve iteracije, medtem ko vertikalna os prikazuje število preostalih ur dela. Diagram se osvežuje vsak dan na podlagi ocen preostalega dela na vseh opravilih s seznama iteracije. Slika 2.3. prikazuje primer diagrama neke iteracije.



Slika 2.1.: Primerjava med planirano in dejansko hitrostjo [7]



Slika 2.2.: Diagram *Release burndown chart* na začetku 7. iteracije [7]



Slika 2.3. Primer diagrama *Sprint burndown chart* [7]

2.3.3 Prednosti

Prednosti, ki jih prinaša metodologija Scrum v razvoj programske opreme, so številne. Projekt razpade v zaporedje obvladljivih delov, pri čemer delo lahko napreduje, tudi če zahteve niso stabilne. Metoda prinaša transparentnost vsem udeleženim in zainteresiranim, ki lahko opazujejo napredek na projektu. Med člani razvojne ekipe se izboljša komunikacija. Ker v razvojni skupini ni hierarhije, so vsi člani deležni zaslug za uspeh, tako med delom kot tudi na koncu projekta. Naročnik dobiva posamezne dele rešitve ob dogovorjenih rokih in s tem

sproten vpogled, kako programska oprema deluje. Na ta način se izboljšajo odnosi z naročnikom ter poveča zaupanje in obseg skupnega znanja. Vse to prispeva k vzpostavitvi ozračja, v katerem vsi udeleženi pričakujejo, da bo projekt uspel [6].

2.3.4 Razširnjenost uporabe v sodobnem času

Glede na zadnjo raziskavo State of Agile je Scrum najbolje razširjena agilna metoda. Scrum uporablja 55% anketirancev, kar 73% anketirancev pa uporablja Scrum ali hibridne metode, povezane s Scrumom [36].

Poglavje 3 Predstavitev projekta

V tem poglavju bom opisal zasnovo projekta prenove spletnega portala za zdravo prehrano skozi njegovo poslanstvo, funkcijo in glavne module. Opisal bom tudi tehnološko zasnovo tako spletne strani kot samega projekta.

3.1 Projekt prenove spletnega portala za zdravo prehrano

Obstoječ spletni portal podjetju služi že dobrih 5 let. Podjetju je omogočil:

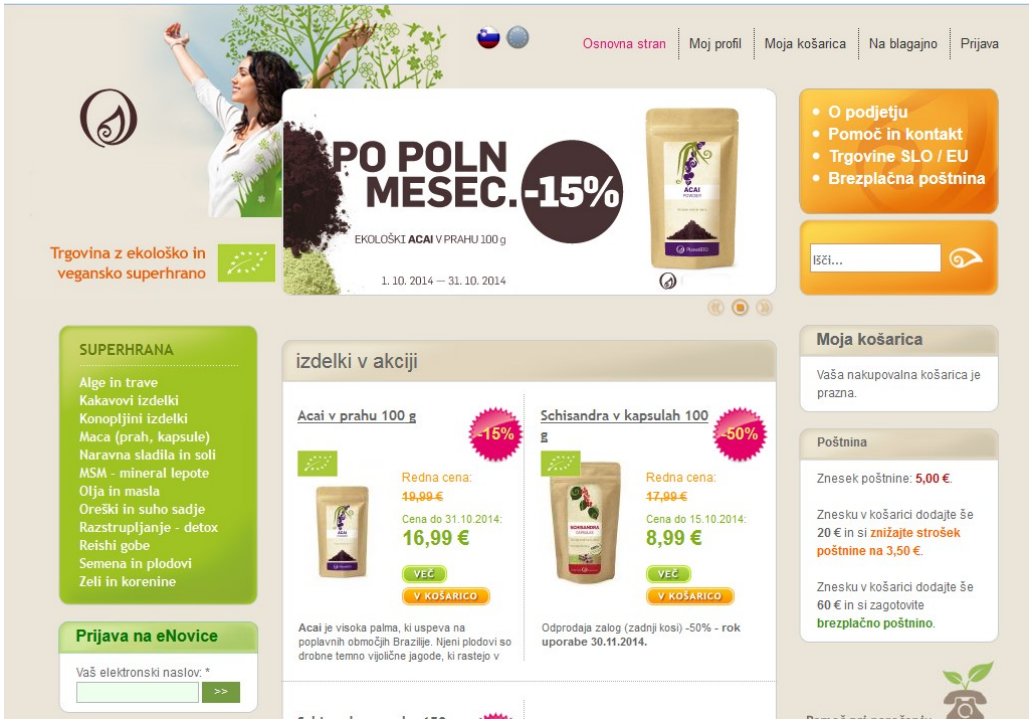
- da je s prejmanjem spletnih naročil začelo uspešno poslovati in rasti,
- da se je na slovenskem trgu uveljavilo z lastno prepoznavno znamko izdelkov,
- da se je (z različico za tujino) začelo uveljavljati na tujih tržiščih,
- da svojim strankam ponudi zanimive in poučne vsebine o zdravi prehrani.

Ker pa se svetovni splet in spletne tehnologije nenehno spreminjajo in posodabljajo, prav tako pa tudi trendi v oblikovanju in dostopanju do spletnih strani, je nastopil čas, da se v skladu s tem prenovi tudi obstoječ spletni portal podjetja.

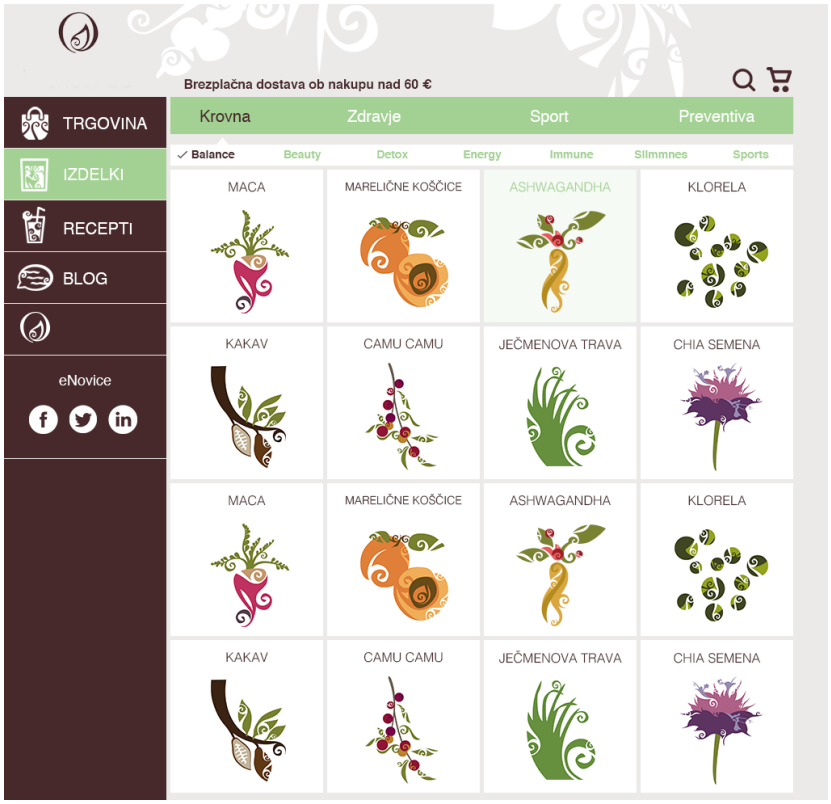
Glavni nameni prenove spletnega portala podjetja za zdravo prehrano so bili naslednji:

- prenovitev izgleda in oblike v skladu s sodobnimi trendi in napravami, ki dostopajo do svetovnega spleta,
- prenovitev arhitekture v skladu s sodobnimi tehnologijami,
- migracija portala na novejšo verzijo nosilne platforme Magento,
- poenotenje obstoječih spletnih strani podjetja v eno pregledno, uporabniku prijazen spletni portal,

- distributerjem / franžizam v tujini ponuditi spletni portal, ki bo skupaj z ustrezno dokumentacijo čim bolj enostaven za namestitev in lokalizacijo, obenem pa v ustrezni luči predstavil podjetje na tujih trgih.



Slika 3.1: Obstoječi spletni portal



Slika 3.2: Nov spletni portal

Sliki 3.1 in 3.2 predstavljata primerjavo med izgledom obstoječega in novega spletnega portala.

3.1.1 Predstavitev poslanstva in funkcije spletnega portala

Spletni portal predstavlja javni izkaz identitete in dejavnosti podjetja, hkrati pa tudi mesto interakcije s kupci oziroma podporniki. Služiti mora kot glavni komunikacijski most med podjetjem in širšim okoljem, v določenem obsegu pa predstavlja mesto izmenjave kakovostnih informacij oziroma dobrin. Glede na to, da podjetje stremi k temu, da bi nagovorilo širšo množico, pričakuje, da bo postal tudi mesto spletnega srečevanja in izmenjave idej podobno mislečih, torej spletno zbirališče zainteresirane javnosti. Spletni portal torej v sebi združuje gospodarsko dejavnost podjetja (prodajo izdelkov), izobraževanje (široko ponudbo koristnih informacij o izdelkih, njihovi uporabi, kmetovanju, zdravem načinu življenja, pomembnih družbenih iniciativah ipd.) in kritično izmenjavo mnenj med podjetjem in uporabniki.

Ker se podjetje zaradi svoje dejavnosti skuša odmakniti od tradicionalnih metod oglaševanja in v ospredje postavlja ponudbo kakovostnih vsebin, spletni portal ne sme biti osredotočen

izključno na spletno prodajalno. Obiskovalcem in uporabnikom mora ponuditi vizualno privlačno, vsebinsko bogato ter navigacijsko enostavno razumljivo spletno okolje, ki bo ponujalo dovolj razlogov, da obiski ne bodo osredotočeni le na nakupovanje, temveč se bodo obiskovalci na portalu zadrževali tudi, ko jih ne bo vodila zgolj realizacija nakupne odločitve. Sledeč standardom ustrezne uporabniške izkušnje mora biti vsa vsebina jasno in čitljivo strukturirana, spletna prodajalna pa mora biti, kolikor je le mogoče, pregledna, torej razbremenjena nepotrebnega vsebinskega balasta.

Realizacija nakupa mora vsebovati minimalno število potrebnih korakov. Navigacija med posamičnimi razdelki (blog, predstavitev izdelkov, prodajalna …) in vsebinami mora vselej slediti enaki logiki, saj v nobenem koraku nočemo, da obiskovalec ne bi vedel, kako je prišel do določene vsebine in kako se vrniti na katerega koli od razdelkov spletnega portala [5].

3.1.2 Predstavitev glavnih razdelkov portala

Glede na zastavljeno poslanstvo in funkcionalnost smo v podjetju opredelili naslednje glavne razdelke spletnega portala:

- **Spletna trgovina**, ki mora ponujati čim krajši postopek realizacije nakupne odločitve. Razbremenjena mora biti vseh spremljajočih vsebin, ki jih obiskovalec lahko najde pod drugimi razdelki (npr. informacije o izdelkih, aktivnostih podjetja ipd.). Nabor funkcionalnosti mora omogočati popuste in druge prijeme pospeševanja prodaje (npr. izpostavitev akcijskega izdelka). Navigacijska logika mora biti zasnovana tako, da je glavnina toka usmerjena k intuitivni in čim krajši realizaciji nakupa.

- **Predstavitev izdelkov**, ki obsega podatke o sestavinah, bazičnih hranilnih vrednostih, poglavitnih dobrobitih, postopkih pridelave in poreklu izdelkov.

- **Predstavitev možnosti uporabe izdelkov** v kulinariki, kozmetiki in pri drugih življenjskih opravilih (recepti, nasveti, navodila).

- **Blog**, ki naj služi predstavitvi aktualnih in drugih družbeno koristnih aktivnosti podjetja, pa tudi objavi različnih člankov, ki podpirajo promocijske dejavnosti podjetja (kot npr. članki o zanimivih lastnostih določenih produktov). V sklopu korporativnega bloga je lahko vključen tudi razdelek za mnenja uporabnikov. Blog je edini razdelek, ki poleg družabnih omrežij obiskovalcu ponuja neposredno izmenjavo mnenj [5].

3.2 Predstavitev platforme Magento za elektronsko poslovanje

3.2.1 Opis platforme

Magento je hitro rastoč, robustni odprtokodni sistem za upravljanje vsebin (CMS - Content Management System) za elektronsko poslovanje. Magento že v osnovi vsebuje mnoge pametne rešitve, kot so npr.: večplastna navigacija, iskalnik z avtomatskim predizpolnjevanjem, podpora večim jezikom, pametno brskanje, RSS-vir katalog produktov v formatu RSS, označevanje in ocenjevanje produktov, ocenjevanje iskalnih pojmov, ocenjevanje zaznamkov strank, menjalne tečaje, integracijo Googlevega kazala, sporočanje o opuščeni nakupovalni košarici, primerjave med produkti, seznam želja, povečevanje slik produktov, itd. Kot tak je zelo primeren kot ogrodje za postavitev raznolikih spletišč, katerih osnovna dejavnost je elektronsko poslovanje.

Magento se razvija v treh različicah [23]:

- Magento CE (Community Edition) je odprtokodna rešitev, ki jo lahko modificira in dodeluje poljuben uporabnik.
- Magento EE (Enterprise Edition) je v jedru enaka rešitvi CE, vendar je plačljiva in nudi več funkcionalnosti. Namenjena je večjim podjetjem, ki zahtevajo več podpore pri namestitvi, konfiguraciji in uporabi.
- Magento GO pa je rešitev v oblaku, ki vključuje tudi spletno gostovanje.

Spletni portal za zdravo prehrano, o katerem govori pričujoče diplomsko delo, temelji na različici Magento CE.

3.2.1.1 Zgodovina

Prvo verzijo platforme Magento je razvilo in izdalo podjetje Varien Inc. 31. marca 2008. Ime Magento je izpeljava iz prvotnega imena Bento in imena Mage, ki je posvetitev liku čarovnika iz fantazijske igre Dungeons & Dragons.

V letu 2010 je 49-odstotni lastnik Magenta postalo podjetje eBay. Od junija 2011 pa je podjetje eBay 100-odstotni lastnik podjetja [23].

V času nastanka tega diplomskega dela je bila zadnja verzija Magenta CE 1.9.0.1 izdana 15. maja 2014.

3.2.1.2 Arhitektura

Platforma Magento temelji na programskem jeziku PHP [30] in ogrodju Zend Framework [38]. Zend je prav tako odprtokodna, objektno orientirana knjižnica, namenjena razvoju spletnih aplikacij, ki temelji na programskem jeziku PHP 5.

Magento bazira na arhitekturi model-pogled-kontroler (MVC - model-view-controller), kar pomeni, da so njegovi elementi, ki opredeljujejo kodo in oblikovnje, ločeni. Prav tako Magento bazira na konfiguraciji, kar pomeni, da za svoje nastavitve preferenčno uporablja konfiguracijske datoteke kot konvencije poimenovanja.

V Magentu se vse datoteke s programsko kodo, ki temeljijo na isti funkcionalnosti, uvrščajo skupaj v module. Moduli so jedro Magenta. Vsak operacija na spletni strani se vrši preko modula. Moduli služijo kot vsebniki za eno ali več od naslednjih stvari: nastavitve, sheme podatkovne baze, podatki podatkovne baze, objekti za upodabljanje, pomožni programi, podatkovni modeli, kontrolerji operacij.

Moduli se dalje povezujejo v bazene kode *(angl. code pools)*, ki omogočajo varno in enostavno prilagajanje in razširitve osnovne funkcionalnosti. Z uporabo bazenov kode razvijalci zagotavljajo, da se bo domorodna funkcionalnost z novimi različicami osnovnega produkta lahko nadgrajevala in bo na ta način zaščitena pred spremembami, ki jih bo hotel storiti uporabnik platforme.

Magento vsebuje tri bazene kode:

- *Core* (jedro) vsebuje bazo aplikacije. Koda v jedru je zbirka vseh modulov, ki jih razvija in certificira ekipa v matičnem podjetju. Kode v jedru ni priporočljivo spreminjati.

- *Community* (skupnost) je mapa, kjer so zbrane datoteke z razširitvami bazena kode *core*. Večina razvijalcev razširitev sistema Magenta namesti dodelave programske kode v ta bazen z namenom, da se ne prekrivajo z jedrom in lokalnim delom.

- *Local* (lokalno) je zbirka lokalnih prilagoditev originalne kode, ki bo uporabljena samo v specifični instanci Magenta. Te razširitve se nahajajo v mapi local z namenom, da ne motijo nadgradenj v jedru in se obenem razlikujejo od prispevkov razvijalske skupnosti [25].

Poleg osnovnih funkcionalnosti, ki ji nudi Magento, obstaja velik nabor razširitev (angl. *extensions*) osnovne funkcionalnosti, ki jih razvija široka skupnost samostojnih razvijalcev programske opreme ali podjetja in jih nudi na svetovnem spletu tako v brezplačnih kot v

plačljivih različicah. Striktni pogoji uporabe prepovedujejo, da bi razširitve spreminjale kodo v jedru.

Podatki v Magentu se shranjujejo v podatkovni bazi tipa MySQL.

3.2.1.3 Zunanji izgled

Zunanji izgled spletnih strani, ki stojijo na ogrodju Magenta, določa t.i. tema *(angl. theme)*. Tema vsebuje datoteke, ki določajo izgled in funkcionalnost obličja spletne platforme.

Da bi lahko natančno nadzorovali izgled spletne platforme, Magento poleg osnovne teme nudi tudi možnost kreiranja lastne, prilagojene teme. Sorodne teme se povezujejo v oblikovne pakete. Ker vsaka namestitev Magenta nudi upravljanje večih spletnih trgovin hkrati, ima lahko vsaka od teh izbrano različno temo.

Več o razvoju lastne prilagojene teme pa sledi v poglavju 3.2.3.

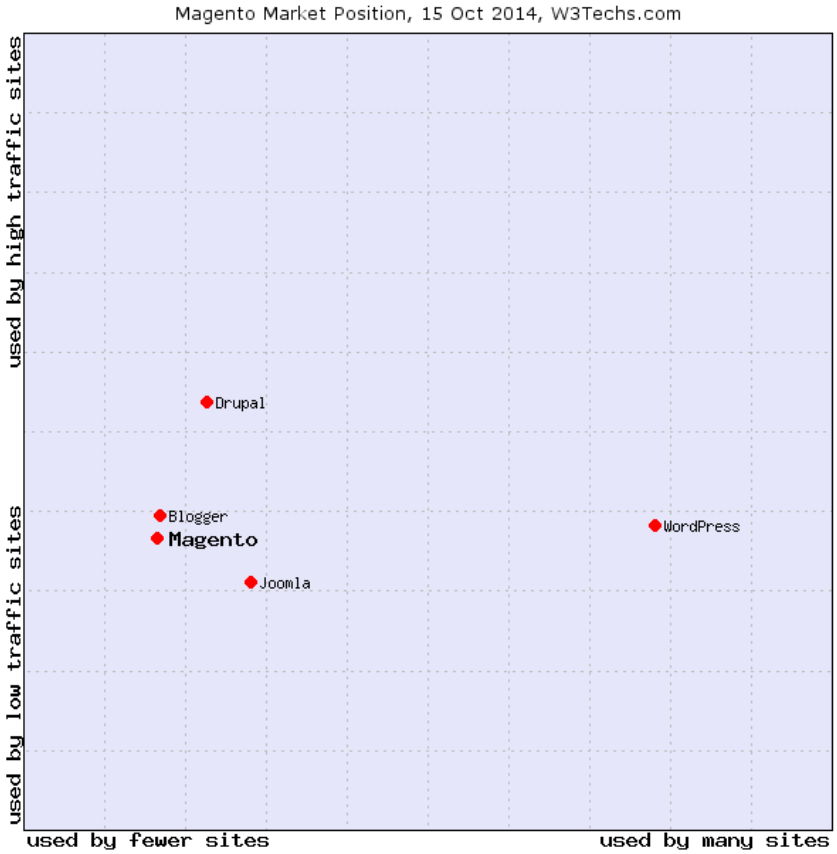
Slika 3.3.: Izgled osnovne teme Magento

3.2.1.4 Statistika uporabe

Po navedbah spletnega portala W3Techs [26] je julija 2014 Magento uporabljalo 2,7% spletnih strani, ki uporabljajo poznane sisteme za upravljanje vsebin. To je kar 1% vseh spletnih strani.

Platforme za elektronsko poslovanje so najhitreje rastoči sektor med sistemi CMS, pri čemer je Magento v vodilni poziciji pred tekmeci, kot so PrestaShop, OpenCart, osCommerce and Shopify.

Poleg Wordpressa je Magento, ki z 2,7% deležem zaseda 5. mesto med CMS-i, najhitreje rastoč sistem v času nastanka te diplomske naloge.



Slika 3.4. Položaj Magenta na trgu CMS [37]

Slika 3.4. prikazuje položaj Magenta na trgu v primerjavi z ostalimi sistemi CMS.

3.2.2 Razlogi za izbor platforme Magento

Kljub temu da Magento uporabniku (poslovnemu subjektu) nudi možnost relativno hitre in cenovno ugodne vzpostavitve platforme za elektronsko poslovanje in s tem umestitev na spletni trg, ni brez slabosti. Naštel bom nekaj najbolj očitnih:

- Uniformiran izgled strani. V poplavi vnaprej pripravljenih tem *(themes)* za Magento se lahko običajnemu uporabniku (potencialnemu kupcu) zdi, da je pristal na isti spletni strani, čeprav gre lahko za drugo spletno trgovino. Problemu uniformiranosti se izognemo z razvojem lastne prilagojene teme, kar pa pripelje do večjih stroškov.

- Počasnost delovanja. Če sistem ni nameščen na ustrezno konfiguriranem strežniku s primernimi zmogljivostmi, redno vzdrževan in pravilno konfiguriran, lahko za odjemalce začne delovati zelo počasi.

- Zapleteno vzdrževanje. Ravno zaradi svoje "robustnosti" Magento v svojem zaledju nudi ogromno možnosti in nastavitev, kar zna biti za povprečnega skrbnika precejšen zalogaj.

- Zapleten razvoj lastnih funkcionalnosti in prilagoditev. Iz izkušenj, pridobljenih v našem podjetju in na dotičnem projektu, lahko napišem, da zaradi svoje kompleksne arhitekture Magento prav tako ni prizanesljiv do razvijalcev, ki niso opravili katerega od uradnih certificiranj, ki jih nudi podjetje Magento. Krivulja učenja zato ni hitro vzpenjajoča, čas razvoja pa se s tem poveča.

Navkljub naštetim slabostim Magento prinaša veliko dobrobiti, kot so:

- v svoji nadzorni plošči omogoča dokaj preprost nadzor in pregled nad nakupi, strankami in izdelki,

- omogoča enostavno realizacijo akcijske ponudbe, popustov, bonov za popuste, ipd.,

- nudi dokaj intuitiven vmesnik za urejanje in objavo poljubnih spletnih strani, anket in elektronskih novic,

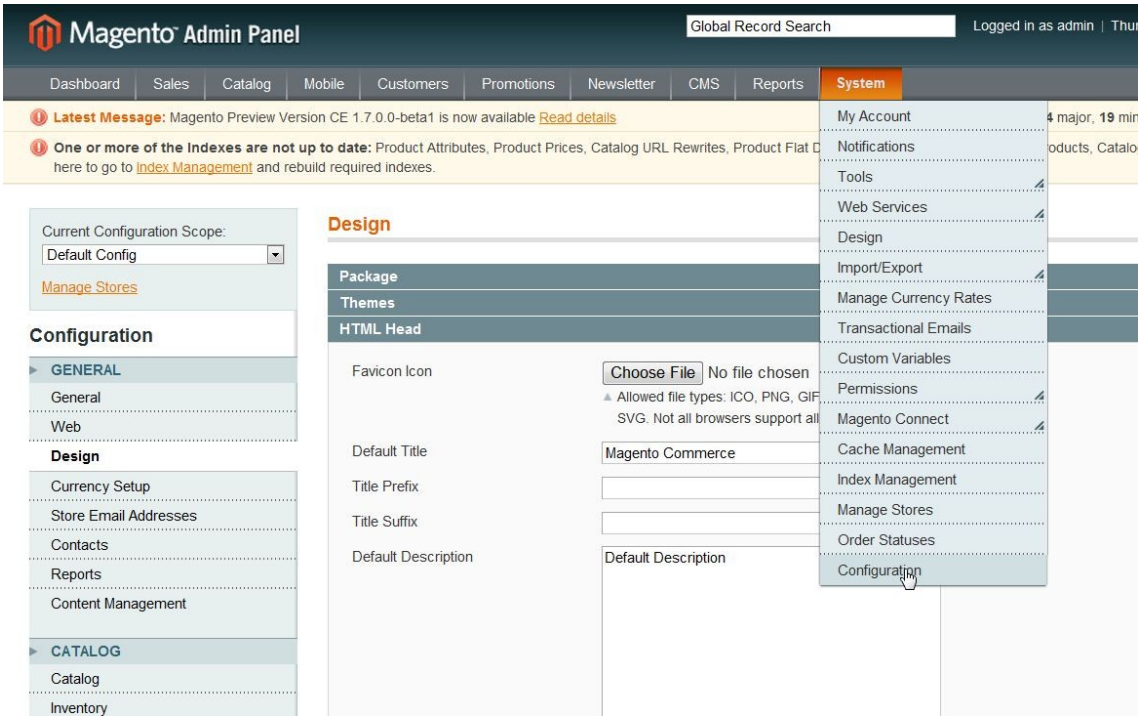
- omogoča izdelavo raznih poročil o prodaji, kupcih, nakupovalnih košaricah, itd.,

- ob primerni implementaciji teme za zunanji izgled nudi obiskovalcem prijetno uporabniško izkušnjo in enostavno nakupovanje.

Poleg naštetih dobrobiti v prid izbora priča tudi znanje in izkušnje o rokovanju s sistemom, ki so se tekom let uporabe nabrale v podjetju. Ključnega pomena za podjetje pa je tudi baza strank in nakupov, ki so se zbrali v času delovanja obstoječega portala, ki prav tako temelji na Magentu, seveda nižje različice. To bazo podatkov bi radi brez večjih zapletov prenesli na novo različico (za kar obstajajo že testirana orodja).

Za tekoče poslovanje v podjetju sta pomembni še dve aplikaciji, in sicer sistem za fakturiranje Vasco in aplikacija Mirko, ki poskrbi za integracijo med Magentom in omenjenim sistemom za fakturiranje (računovodskim programom). Gre predvsem za prenos internetnih naročil iz Magenta v Vasco, prenos stanja plačil naročil iz Vasca v Magento in usklajevanje zalog.

Ker je bila aplikacija Mirko razvita znotraj podjetja prav v namen integracije teh dveh sistemov, smo se v podjetju poleg zgoraj naštetih razlogov pri razvoju novega portala ponovno odločili za Magento, seveda za eno od novejših različic.



Slika 3.5.: Nadzorna plošča sistema Magento

3.2.3 Implementacija prilagojene (custom) teme za Magento

Kot sem že omenil v poglavju 3.2.1.3, za Magento lahko razvijemo lastno temo, ki je zbirka datotek, ki določajo izgled in delovanje čelnega dela sistema. Tema je sestavljena iz različnih

tipov datotek: CSS, slike, za temo specifični JavaScript, XML, PHTML, etc. Te datoteke so shranjene v dveh glavnih direktorijih Magentovega datotečnega sistema:

- *app/design*, kjer se nahajajo datoteke, ki kontrolirajo, kako se izrisujejo predloge in
- *skin*, kjer so datoteke, ki nadzorujejo vizualni aspekt teme (CSS, slike, ipd.).

Nadalje so datoteke teme organizirane v naslednje poddirektorije:

- *layout*: vsebuje osnovne datoteke XML, ki definirajo strukturo blokov za različne strani in kontrolne meta podatke,
- *template*: vsebuje datoteke PHTML, ki vsebujejo oznake xHTML in bloke v jeziku PHP, ki so potrebni za kreiranje logike vizualne prezentacije,
- *locale*: vsebuje preproste tekstovne datoteke CSV, ki vsebujejo prevode za nize znakov (v obliki parov ime-vrednost), porazdeljene v poddirektorije po različnih jezikih [25].

Prednosti lastne prilagojene teme so unikatna vizualna izkušnja spletnega mesta, prilagojen uporabniški vmesnik, ki ustreza vsebini spletnega portala, realizacija intuitivne, uporabniku prijazne navigacije in nakupovanja. Tako je večina dela na tem projektu usmerjena v razvoj lastne teme.

3.3 Izbor razvojnih orodij

Zmanjševanje stroškov razvoja botruje izboru večinoma odprtokodnih razvijalskih orodij in tehnologij. Ker je obseg projekta klical vsaj po dveh razvijalcih, je bil potreben tudi izbor sistema za nadzor verzij.

3.3.1 Sistem za nadzor verzij Git in spletna storitev Bitbucket

Git je sistem za porazdeljeno dokumentiranje sprememb in izdajanje različic datotek s poudarkom na hitrosti, celovitosti podatkov in podpori porazdeljenih, nelinearnih delovnih tokov. Git je prvotno zasnoval in razvil Linus Torvalds za razvoj jedra operacijskega sistema Linux leta 2005. Od takrat je postal najširše povzet sistem za nadzor verzij pri razvoju programske opreme. [18]

Ker je razvojna ekipa v našem podjetju majhna, smo za uporabo Gita izbrali najpreprostejši centralni delovni tok, čeprav Git glede na potrebe različnih razvijalskih ekip in projektov nudi tudi bolj kompleksne implemetacije delovnih tokov.

Centralni delovni tok je zelo podoben tistemu, ki ga implemtira SVN, znani Apache-ov sistem za nadzor verzij, naslednik široko uporabljenega sistema CVS. Kljub temu pa Git prinaša nekaj prednosti pred SVN.

Prvič vsakemu razvijalcu nudi lokalno kopijo celotnega projekta. To izolirano okolje omogoča razvijalcu, da dela neodvisno od ostalih sprememb na projektu, vse dokler se mu lokalna verzija ne zdi primerna za pošiljanje v centralni repozitorij. Drugič pa Git nudi robusten model zlivanja in podvej, ki je varen mehanizem za integriranje kode in delitev sprememb med repozitoriji. [15]

Ker smo v podjetju zaradi distribuiranosti ekipe potrebovali dostop do repozitorija s kodo praktično povsod, kjer je na voljo internet, in repozitorija zaradi varnostnih razlogov nismo hoteli gostiti znotraj omrežja podjetja, je bila logična odločitev za spletno storitev, ki bi omogočala gostovanje in enostaven dostop do odložišča programske kode sistema Git.

Izbrali smo spletno storitev Bitbucket (https://bitbucket.org/) podjetja Atlassian, ki je brezplačna za do 5 uporabnikov in nudi varen dostop preko protokola HTTPS. Poleg tega za razliko od spletne storitve GitHub nudi možnost privatnih repozitorijev, torej koda projekta ni javno dostopna.

3.3.2 Uporaba orodij IDE

Na projektu uporabljamo različna integrirana razvojna okolja (IDE - Integrated Development Environment), ki so primerna za urejanje programske kode: PHP, CSS, HTML, XML, JavaScript, ipd. Med njimi so NetBeans, PhpStorm pa tudi preprost urejevalnik Notepad++. Tukaj je izbira prepuščena razvijalcem samim.

Poglavje 4 Prikaz uvajanja metode Scrum na projektu

4.1 Priprava in uvajanje metode Scrum na projektu

Potem ko so se v podjetju z željo po prenovi spletnega portala in formiranju dokumenta "Predlog konceptualne opredelitve prenovljene spletnega portala" [5] začele priprave na projekt. Pri izboru metodologije dela smo uporabili znanje, pridobljeno pri predmetu Tehnologija programske opreme, kjer smo se seznanili z agilnim razvojem programske opreme.

Ravno to novo znanje, ki smo ga dobili pri predmetu, in dejstvo, da se je v podjetju pričenjal projekt, ki v štartu ni bil popolnoma dorečen in je kazal na to, da se bodo zahteve med razvojem še spreminjale, sta pripeljala do odločitve, da bi bila metoda Scrum zelo primerna za vodenje tega projekta.

K tej odločitvi je dodatno pripomoglo dejstvo, da je projekt navkljub nepopolni zasnovi potreboval zagon, saj je bila prenova portala izražena kot ena glavnih prioritet v podjetju. Poleg tega pa so se že nekako "naravno" formirale vloge, ki so precej spominjale na tiste iz metodologije Scrum.

Vse našteto je dalo motivacijo, da metodo Scrum skušamo vpeljati v projekt z namenom, da čim bolje formaliziramo razvoj produkta, čeprav je bilo jasno, da se vseh priporočil metode zaradi situacije v podjetju ne bomo mogli držati.

4.1.1 Formiranje ekipe in delitev vlog

Ker gre za manjše podjetje (ca. 15 zaposlenih), na projektu prenove spletnega portala ni bilo pričakovati velike ekipe, smo pa za čas projekta najeli dodatno razvijalko spletnih rešitev.

Vlogo lastnika izdelka (angl. Product Owner) sta prevzela dva zaposlena, in sicer avtor dokumenta konceptualne opredelitve in glavni razvijalec stare verzije portala. Torej je šlo za naročnike znotraj podjetja. Poleg opredelitve sta lastnika izdelka poskrbela za izdelavo skeletnih skic novih spletnih strani in prevzela komunikacijo z oblikovalko, ki je na podlagi skeletnih skic izdelala datoteke v Photoshopu (PSD - Photoshop Document) z oblikovno zasnovo za večino strani.

Na žalost sta lastnika izdelka, zadolžena kot vodja procesov in vodja prodaje, po začetnem vložku zaradi vpletenosti v številne naloge v podjetju kasneje v projektu razvoja lahko sodelovala le malo, kar se je, kot bom opisal kasneje, pokazalo za enega od razlogov, da projekt ni potekal po načrtih.

Vlogo razvojne skupine (Team) sva prevzela avtor tega diplomskega dela in dodatno najeta razvijalka, ki je zaradi druge zaposlitve v povprečju lahko nudila tri dni na teden za razvoj na našem projektu. Prav tako nisem mogel posvetiti celotnega tedna razvoju novih funkcionalnosti, saj sem zadolžen tudi kot skrbnik starega portala. Tudi to je botrovalo počasnejšemu razvoju.

Kot pobudnik in nosilec znanja metodologije Scrum sem prevzel tudi vlogo skrbnika metodologije (Scrum Master).

Majhnost ekipe in prekrivanje vlog tako ni dopuščalo strogega sledenja metodologiji in je klicalo k dobršnji meri improvizacije. To pa seveda pripomore k manjši predvidljivosti rezultatov. K temu je dodatno prispevalo dejstvo, da člana razvojne skupine navkljub poznavanju bazičnih tehnologij (PHP, CSS, HTML in JavaScript) za sabo še nisva imela predhodnega večjega projekta na platformi Magento.

4.1.2 Izbor aplikacije za projektno vodenje

V času priprave projekta, medtem ko je oblikovalka v navezi s lastniki izdelka pripravljala grafično zasnovo, sem raziskal ponudbo orodij za vodenje projektov v skladu z metodologijo agilnega razvoja, predvsem tiste, ki so nudile podporo metodi Scrum. Ta naloga se je izkazala za težjo in dolgotrajnešo od pričakovane. Vodila pri izbiri orodja so bila naslednja:

- implementacija, ki se čim bolje drži načel metode Scrum, kot smo se je učili na predavanjih, a hkrati dopušča fleksibilnost,
- pregleden in intuitiven grafični vmesnik, ki po možnosti vključuje simulacijo table s karticami,
- možnost gostovanja aplikacije pri samem ponudniku, ki skrbi za varnostne kopije in podporo,
- možnost dostopa preko svetovnega spleta in varnih povezav zaradi distribuiranosti ekipe.

Testiral sem naslednje aplikacije, ki bi lahko služile vodenju projekta:

- Mingle podjetja ThoughtWorks [29], ki se je izkazal kot pregleden in zelo fleksibilen z možnostjo izbora različnih predlog za različne projekte (Kanban, Agile, Scrum, itd.), vendar pa so bile ravno te predloge projektov s svojimi implemetacijami delovnih tokov in sistem kartic prezapletene za naše potrebe.

- AgileFant avtorja Jarna Vähäniittya [11] je imel nekoliko okorn in neintuitiven uporabniški vmesnik za ravnanje s seznamom nalog in iteracijami. Prav tako ni nudil table s karticami.

- Agilo for Scrum podjetja Agilo Team [13] je pri testiranju dokaj pogosto javljal nepričakovane napake in ni nudil grafov za spremljanje napredka.

- Rally podjetja Rally Software [32], kjer nas je zmotilo dejstvo, da se na opravilih ni dalo beležiti opravljenih ur po dnevih, pač pa samo začetno oceno predvidenega časa za realizacijo in oceno, koliko ur je še za opraviti na opravilu.

- JIRA s svojo razširitvijo za agilne metode JIRA Agile podjetja Atlassian [20] se je s svojo dodelanostjo grafičnega vmesnika, intuitivnostjo uporabe, grafi za spremljanje napredka in vsemi potrebnimi parametri za vodenje projekta po metodi Scrum najbolje približala naši želji po aplikaciji, ki bi jo radi uporabljali na dnevni bazi tekom razvoja. K izboru je pripomogla tudi dokaj ugodna cena za ekipe do 10 članov pa tudi moje izkušnje pri delu z aplikacijo JIRA (brez dodatka za agilni razvoj) v prejšnjem podjetju. Zaradi delne distribuiranosti ekipe smo se odločili za verzijo v oblaku.

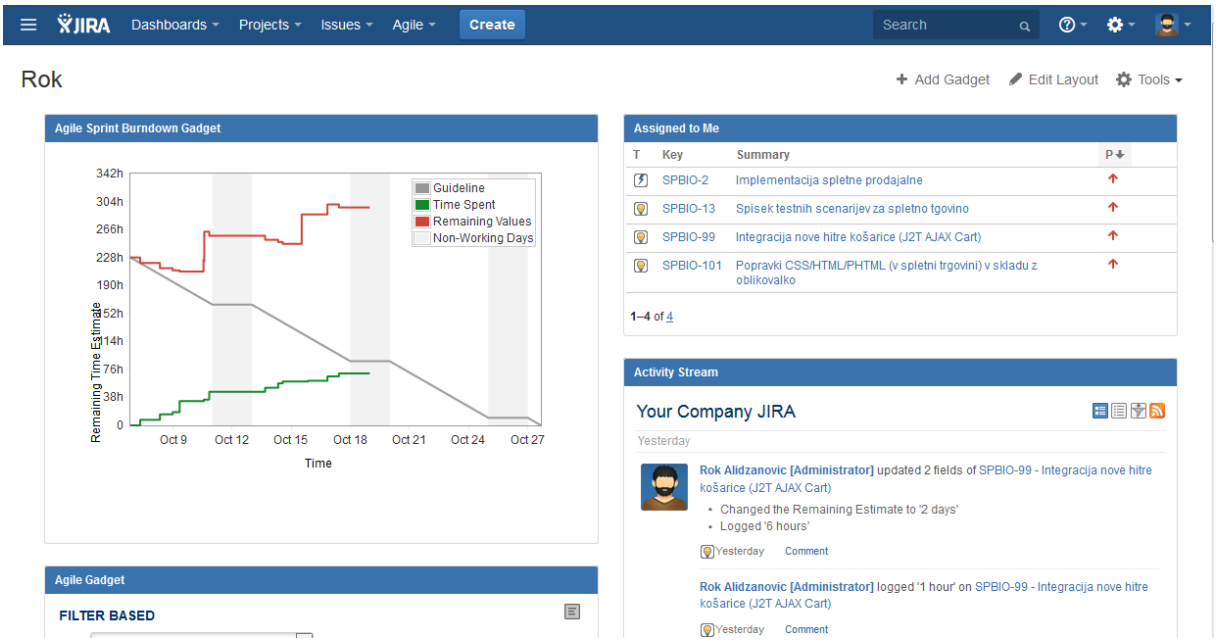
4.1.3 Opis izbrane aplikacije za projektno vodenje: JIRA

4.1.3.1 Splošen opis

JIRA je zakonsko zaščitena programska oprema podjetja Atlassian za sledenje zadev (issue). JIRA nudi funkcije za vodenje projekta, sledenje programskih napak - hroščev in drugih zadev na projektu. Čeprav se črkuje z velikimi črkami, ime ni kratica pač pa okrajšava za Gojira, japonskega imena za pošast Godzilo. Razvija se od leta 2002.

JIRA je napisana v programskem jeziku Java in omogoča povezavo z naslednjimi programi za nadzor verzij: Subversion, CVS, Git, ClearCase, Team Foundation, Mercurial on Perforce. Njene izdaje vključujejo podporo različnim jezikom, kot so: angleščina, japonščina, nemščina, francoščina, španščina in druge. Glede na podatke podjetja Atlassian JIRO

uporablja več kot 25 tisoč strank v 122 državah po svetu. Poleg plačljivih licenc podjetja podjetje nudi brezplačno verzijo za določene odprtokodne projekte, neprofitne in neakademske organizacije. [21]



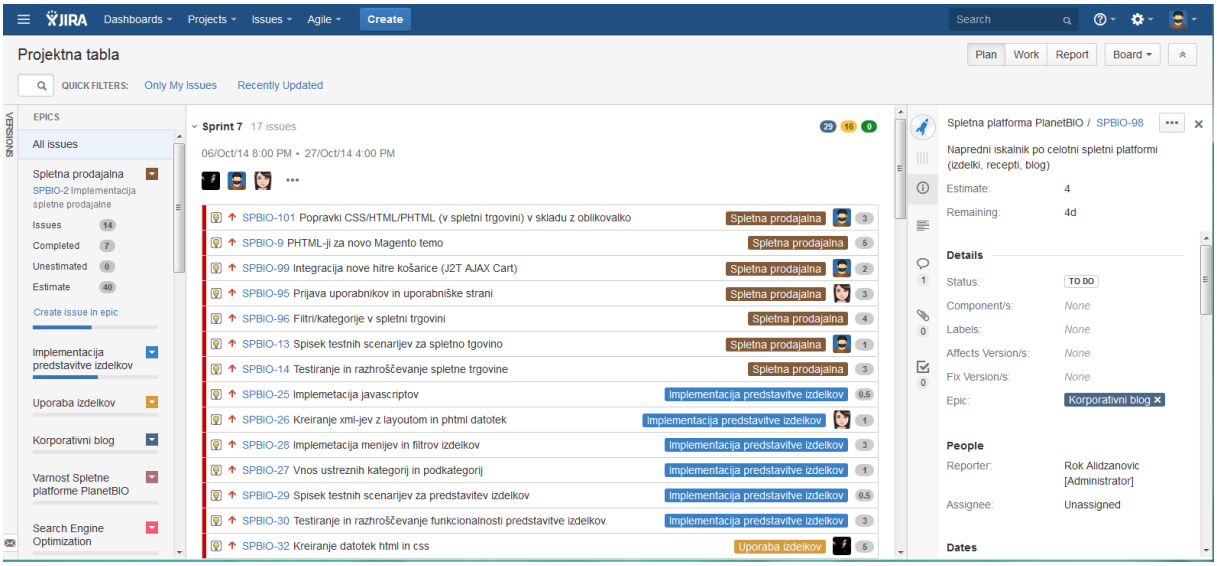
Slika 4.1.: Pregledna plošča v aplikaciji JIRA

4.1.3.2 JIRA Agile

JIRA Agile je fleksibilno orodje za urejanje seznama nalog (Sprint Backlog), planiranje iteracij (Sprint), upravljanje vsakodnevnega dela, sledenje napredku in komunikacijo z naročniki.

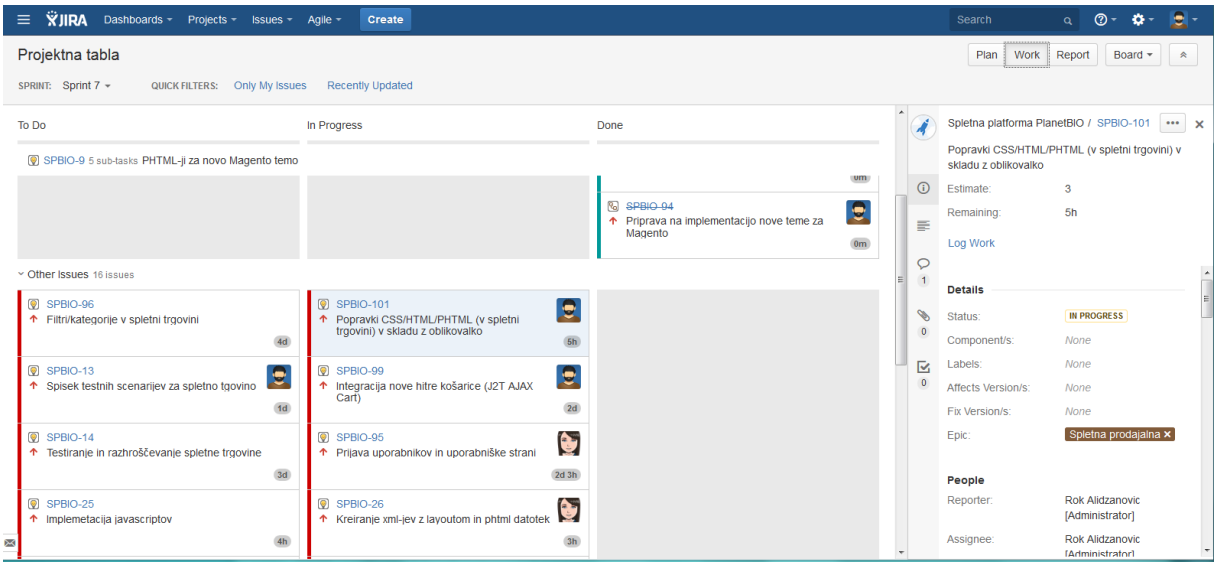
JIRA Agile omogoča delo z virtualnimi karticami in projektnimi tablami. Kartice na projektni tabli delujejo po principu primi in potegni. Te (distribuiranim) ekipam nudijo ažuren vpogled v stanje na projektu, možnost spreminjanja statusa opravil na tabli (*To Do* - čaka na realizacijo, *In Progress* - v delu in *Done* - narejeno) in obenem omogočajo poročanje o opravljenem delu, popravke ocen do konca opravila, komentarje, ipd.

Projektna tabla v JIRA Agile ima tri glavne poglede - načine delovanja. *Plan* (planiranje), ki ga vidimo na sliki 4.2., služi vzdrževanju seznama nalog in planiranju iteracij. Na levi je prikazan pogled realizacije po velikih uporabniških zgodbah (Epic), v osrednjem delu je prikazan seznam nalog po sprintih in nižje preostanek, ki ni bil še razvrščen v sprinte. Na desni je omogočen detajlni pogled posamezne kartice, na katero kliknemo. Glede na vloge metode Scrum naj bi največ služil produktnemu lastniku.



Slika 4.2.: Pogled *Plan* na projektni tabli

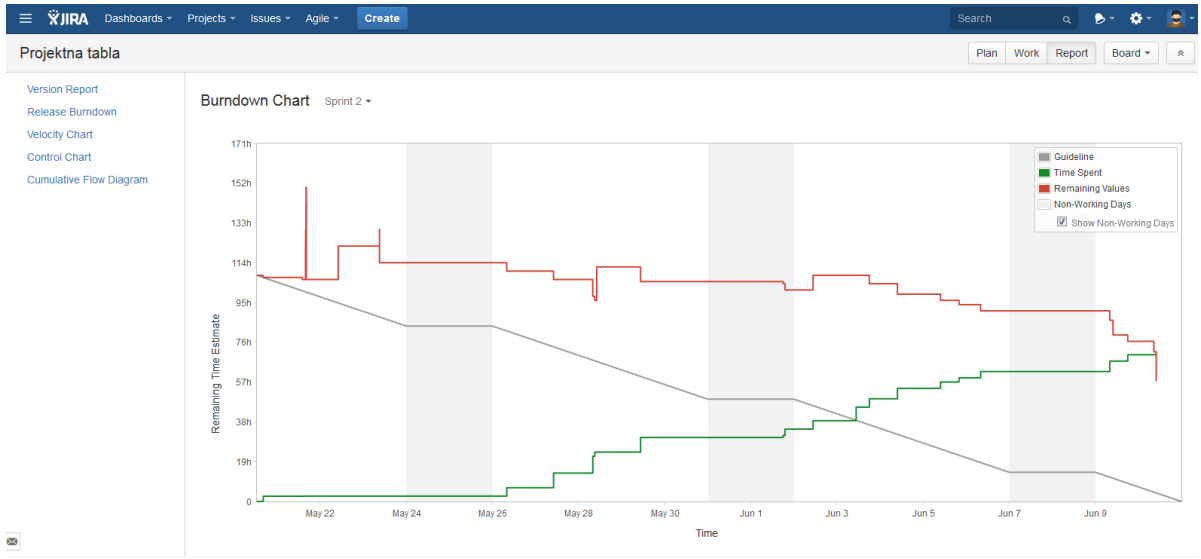
Način *Work* (delo), ki ga vidimo na sliki 4.3., služi vpogledu v delo na trenutno aktivnem sprintu. Glede na vloge metode Scrum v največi meri služi razvojni skupini. Ko gre določeno opravilo v izvedbo, njegovo kartico v načinu primi in potegni prestavimo v iz statusa *To Do* (čaka na realizacijo) v status *In Progess* (v izvedbi). Ko pa je opravilo zaključeno, se ga na enak način prestavi v status *Done* (zaključeno).



Slika 4.3.: Pogled *Work* na projektni tabli

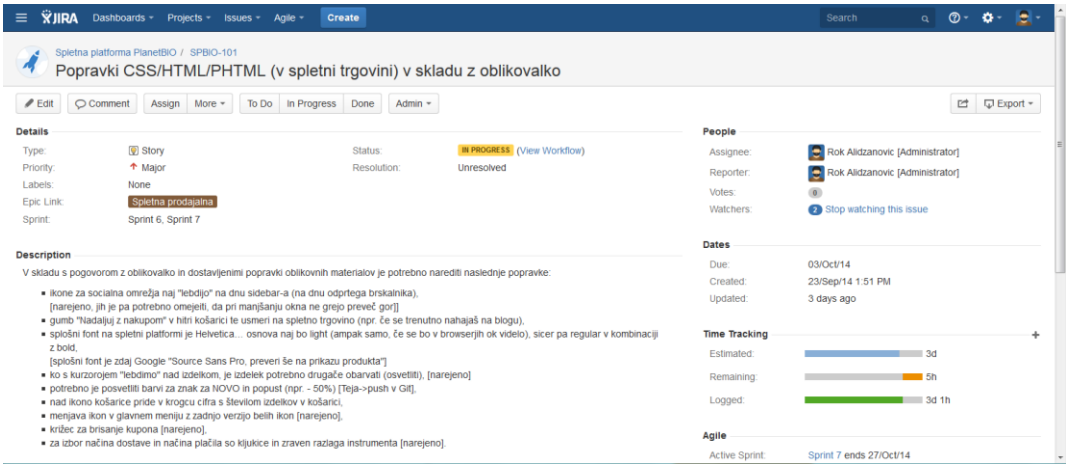
Na desni strani načina *Work* lahko vidimo podrobnosti posamezne izbrane kartice in beležimo opravljeno delo.

Način *Report* (poročanje), prikazan na sliki 4.4., z različnimi grafi služi spremljanju napredka projekta v posameznih iteracijah in v celoti. Tako lahko služi vsem udeležencem projekta. Skrbniki metodologije in lastniki izdelka s pomočjo poročil spremljajo napredek razvijalske ekipe in identificirajo ovire in "ozka grla" na projektu. Razvijalska ekipa pa s pomočjo *Burndown* diagrama spremlja, kako napreduje proti ciljem trenutne iteracije.



Slika 4.4.: Pogled *Report* na projektni tabli

Posamezno zadevo/uporabniško zgodbo/opravilo na projektni tabli lahko izberemo za podroben, razširjen pogled. V tem načinu lahko urejamo njen tip, prioriteto, status, komu je dodeljena, opis, komentarje, idr. Podroben pogled lahko vidimo na sliki 4.5.



Slika 4.5.: Podroben pogled kartice

4.1.4 Izzivi s porazdeljenostjo ekipe

Kot sem že omenil, smo nekaj tehničnih ovir z delno porazdeljenostjo ekipe rešili z izborom sistema za nadzor verzij Git in spletno storitvijo Bitbucket, ki omogoča hranjenje centralnega repozitorija programske kode in drugih datotek v oblaku. Prav tako je določene ovire rešila JIRA: npr. beleženje opravljenega dela, virtualna projektna tabla namesto fizične za spremljanje dela na projektu.

Poleg teh pa je bilo potrebno rešiti tudi komunikacijske ovire. Za reševanje teh smo v času odsotnosti članov, ki niso delali v prostorih podjetja, uporabljali elektronsko pošto, telefonske pogovore in aplikacijo Skype. Te kanale smo uporabili predvsem, ko je pri razvoju prišlo do težav.

Kot skrbnik metodologije na projektu sem mnenja, da bi delo na projektu potekalo bolj tekoče, če bi razvojna ekipa lahko ves čas projekta sodelovala v istem prostoru, saj je na tak način komunikacija med člani hitrejša, bolj neposredna, rešitve problemov pa se najdejo hitreje. Prav tako je lažje bolj dosledno izvajati priporočila metodologije Scrum. Na žalost stanje v podjetju in dogovori z zunanjimi sodelavci tega niso dopuščali.

4.1.5 Oblikovanje seznama opravil

Seznam opravil (angl. Product Backlog) je nastal predvsem na podlagi dokumenta s predlogom konceptualne opredelitve prenove spletnega portala [5] in zaslonov z obliko, ki jih je pripravila oblikovalka. Pripravo seznama sem prevzel skrbnik metodologije, čeprav bi na tem mestu pripomogla tudi angažiranost produktnih lastnikov, ki pa so se, kot sem že omenil, v tej točki lahko le še omejeno udejstvovali na projektu.

Seznam opravil sem pripravil tako, da sem podrobno prečesal dokument konceptualne opredelitve ter zaslone z obliko in izluščil glavne uporabniške zgodbe. To so bile večinoma opredelitve glavnih modulov spletnega portala in nekaterih večjih sklopov opravil.

Te sem nato razdelal na manjša opravila, ki so izhajala iz teh večjih uporabniških zgodb in so bila že bolj namenjena razvojni ekipi, ki je na podlagi le-teh pripravila ocena za realizacijo. Priprava ocen je opisana v naslednjem poglavju.

Tako je nastal dokument Projekt implementacije novega spletnega portala: Product Backlog (začetna zbirka uporabniških zgodb) [1] s seznamom opravil. Seznam opravil je na začetku obsegal 21 uporabniških zgodb, v katerih so nastopale naslednje uporabniške vloge: kupec v spletni trgovini, obiskovalec spletnega portala, skrbnik spletnega portala, uporabnik v oddelku odpreme naročil.

Uporabniška zgodba	**Ocena (št. točk)**
Kot obiskovalec spletnega portala želim **predstavitev izdelkov**, ki obsega podatke o bazičnih hranilnih vrednostih, poglavitnih dobrobitih, postopkih pridelave in proizvajalcih.	21
Kot obiskovalec spletnega portala želim **predstavitev možnosti uporabe izdelkov** v kulinariki, kozmetiki in pri drugih življenjskih opravilih**: recepti,** nasveti, navodila, ipd.	25,5
Kot obiskovalec želim na spletnem portalu predstavitev aktualnih in drugih širše družbeno koristnih aktivnosti podjetja, vključno s predstavitvami tekočih promocijskih dejavnosti v obliki **korporativnega bloga**.	7

Tabela 4.1.: Primeri uporabniških zgodb

Tabela 4.1. prikazuje nekaj primerov uporabniških zgodb in ocen v točkah.

Podajam primer razbitja uporabniške zgodbe, ki je služila kot opredelitev modula za predstavitev možnosti uporabe izdelkov, na opravila:

> *"Kot obiskovalec spletnega portala želim **predstavitev možnosti uporabe izdelkov** v kulinariki, kozmetiki in pri drugih življenjskih opravilih: **recepti**, nasveti, navodila, ipd.* ***25,5t***
>
> - *kreiranje datotek HTML in CSS za postavitev oblike modula recepti: 5t,*
> - *vnos ustreznih kategorij in podkategorij receptov: 1t,*
> - *implemetacija menijev in filtrov receptov: 2t,*
> - *nakup (in prilagoditev) ali implemetacija modula za vnos in prikaz receptov: 5t,*
> - *implemetacija forme za oddajo receptov obiskovalcev strani (na gumbu "Oddaj svoj recept"): 3t,*
> - *backend modul za pregled oddanih receptov obiskovalcev pred njihovo objavo: 3t,*
> - *implemetacija gumba "Dodaj uporabljene izdelke v košarico": 3t,*
> - *spisek testnih scenarijev za testiranje receptov: 0,5t,*
> - *testiranje in razhroščevanje funkcionalnosti receptov: 3t" [1].*

Opravila smo ocenili s točkami, ker so bila nekatera še vedno obsežna. Več o pripravi ocen uporabniških zgodb pa v naslednjem poglavju 4.1.6.

Skupen seštevek ocen uporabniških zgodb s seznama opravil je bil naslednji:

- spletni portal za slovenski trg: 109 točk,
- prilagoditev portala za tujino: 23,5 točk.

Ob takratni predpostavki hitrosti razvoja 7 točk (več o določitvi hitrosti v poglavju 4.2.2) na teden je bilo ocenjeno trajanje projekta:

- spletni portal za slovenski trg: približno 16 tednov,
- prilagoditev portala tujino: približno 4 tedne.

Po izboru aplikacije za vodenje projekta (JIRA) sem uporabniške zgodbe, opravila in ocene vnesel v seznam nalog aplikacije.

4.1.6 Priprava ocen uporabniških zgodb

Ko sem uporabniške zgodbe razbil na manjša opravila, so bila ta pripravljena za ocenjevanje zahtevnosti v točkah iz strani razvojne skupine. Ker sva bila v razvojni ekipi samo dva člana, sva uporabila poenostavljeno verzijo metode *Planning Poker* [3]. Vsak od naju je vzel dokument, šel čez uporabniške zgodbe in opravila in zraven pripisal svojo oceno v točkah. Ko sva se sestala naslednjič, sva šla čez vse ocene in še enkrat prediskutirala obseg vseh nalog, še posebej tistih, kjer so se najine ocene razlikovale. Nato sva se odločila, ali bo obveljala nižja ali višja ocena ali pa neka vmesna vrednost.

Pri ocenjevanju s točkami sva se držala Fibonaccijeve lestvice (1/2, 1, 2, 3, 5, 8, 13, 20, 40, 80), ki se pogosto uporablja pri ocenjevanju uporabniških zgodb po metodi *Planning Poker*.

Kot je napisano v dokumentaciji JIRE Agile [31], imajo tudi najboljši inženirji težave pri napovedovanju prihodnosti. To opravilo ocenjevanja nalog je bilo za naju s sodelavko še toliko težje, ker nisva imela veliko izkušenj z razvojem večjih funkcionalnosti za Magento.

4.2 Potek projekta in merjenje napredka

4.2.1 Določitev trajanja iteracije

Tipična dolžina iteracij na projektih po metodologiji Scrum je 1 do 4 tedne. Na našem projektu smo izbrali dolžino iteracije (sprinta) tri tedne. V razvijalski ekipi sva bila namreč

dva člana, ki nisva mogla vseh delovnih ur nameniti samo temu projektu in smo tako ocenili, da je 3 tedna najmanjše časovno obdobje, v katerem bi lahko zaključili nek sklop funkcionalnosti. Po drugi strani pa nismo hoteli daljših iteracij, da bi lahko hitreje ukrepali v primeru neuspešnih iteracij, kar pa nam kljub temu ni najbolje uspevalo, kot bomo videli kasneje.

Po priporočilih metodologije naj bi bile iteracije konstantne, vendar smo pri nekaterih dolžino skrajšali ali podaljšali za kak dan. To se je zgodilo v primerih, ko sva se z drugo članico razvojne ekipe, ki je v prostorih podjetja delala enkrat do dvakrat na teden, ob koncu sprinta zaradi prikladnosti dogovorila za kak drug dan srečanja na sedežu podjetja kot natančno po treh tednih.

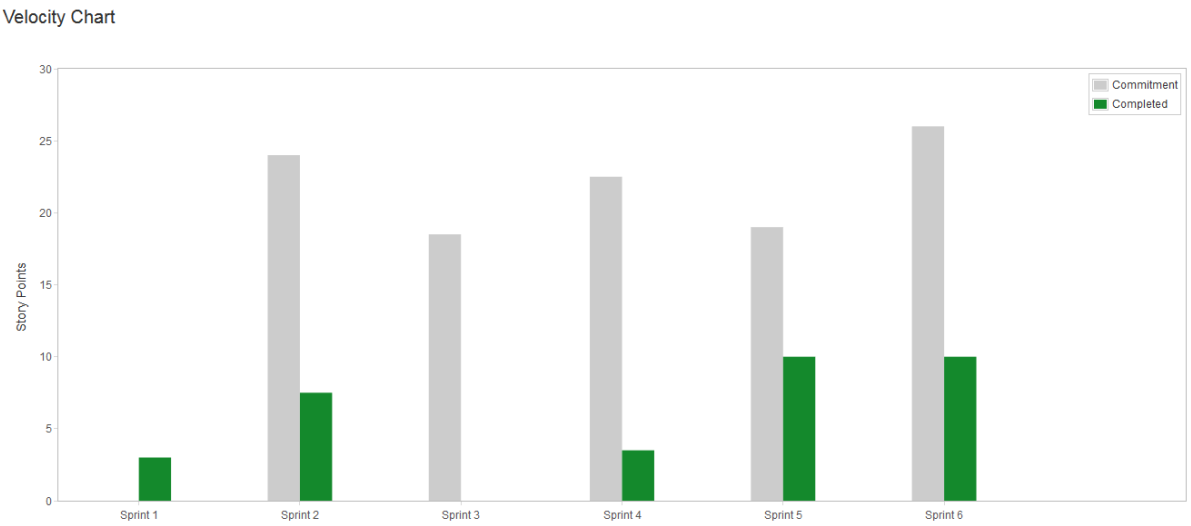
V času poletnih dopustov pa smo en sprint podaljšali na skoraj dvojno dolžino približno šestih tednov z namenom, da bi v tem časovnem obdobju opravili približno enako število ur (točk) kot v normalni iteraciji.

4.2.2 Določitev hitrosti in planiranje iteracij (sprintov)

Pri določanju hitrosti smo za dolžino 1 točke izbrali 1 delovni dan, ta pa je, upoštevajoč privzeto nastavitev v JIRI, 8ur. Na ta način smo v oceni 1 točke upoštevali tudi vse motnje, ki so prisotne med delovnim dnem. Tako je 1 točka postala enakovredna enemu človek-dnevu.

Samo hitrost razvoja sva s sodelavko določila pred vsako naslednjo iteracijo na podlagi dogovora, koliko točk (človek-dni) bo kdo lahko v naslednjih (običajno) treh tednih namenil razvoju na danem projektu. To je običajno znašalo 4,5 mojih točk in 3 sodelavkine točke na teden, kar je skupaj naneslo 22,5 točk.

Na podlagi tako ocenjene hitrosti sva pri planiranju naslednje iteracije med nedokončanimi opravili izbrala ustrezen nabor nalog, katerih skupna ocena točk je približno ustrezala tej oceni hitrosti. Kot vidimo iz grafa na sliki 4.6., pa se je dejanska hitrost, izračunana na podlagi zaključenih opravil, razlikovala od ocenjene.



Slika 4.6.: Primerjava planirane in dejanske hitrosti razvoja po sprintih

Na sliki 4.6. sivi stolpec prikazuje planirano hitrost, zeleni pa dejansko hitrost po iteracijah, ki je precej manjša od planirane. Ti podatki kažejo na to, da bi pri planiranju vsake naslednje itaracije morala upoštevati tudi zgodovinske podatke, vendar pa sva v upanju, da se bo razvoj zaradi pričakovane krivulje učenja le pohitril, vztrajala pri prvotni metodi načrtovanja hitrosti. K temu je prispevala tudi želja naročnikov po čim prejšnjem končnem rezultatu.

4.2.3 Opis tipične iteracije na projektu

Vsaka nova iteracija se je začela z zaključkom prejšnje. Kot sem že napisal, sva se zadnji dan sprinta s sodelavko sestala v prostorih podjetja in v prvi polovici delovnega časa opravljala delo na nezaključenih opravilih. V drugi polovici delovnika sva pregledala kandidate opravil za zaključek in presodila/testirala, ali so dejansko zaključena po načelih principa *Done*. Na preostalih opravilih sva po potrebi popravila ocene do zaključka in nato zaključila tekoči sprint. To del sestanka je po metodologiji Scrum ustrezal sestanku Sprint Review Meeting. Tukaj bi seveda pomagala aktivna vloga lastnikov produkta, ki sva jo v teh primerih prevzela člana razvojne ekipe.

Ob zaključku spinta sva se pogovorila o tem, kaj je šlo v tem sprintu dobro in kaj bi bilo lahko bolje. Tako je ta sestanek služil tudi kot neformalni približek sestanku *Sprint Retrospective Meeting*, ki ga v bolj formalni obliki nismo izvajali.

Po zaključku sprinta je sledilo planiranje naslednje iteracije ali tako imenovani Sprint Planning Meeting. V tem delu sva na podlagi planirane hitrosti in določenih prioritet izbrala v prejšnjem sprintu nedokončane naloge in s seznama nalog (angl. Product Backlog) opravila,

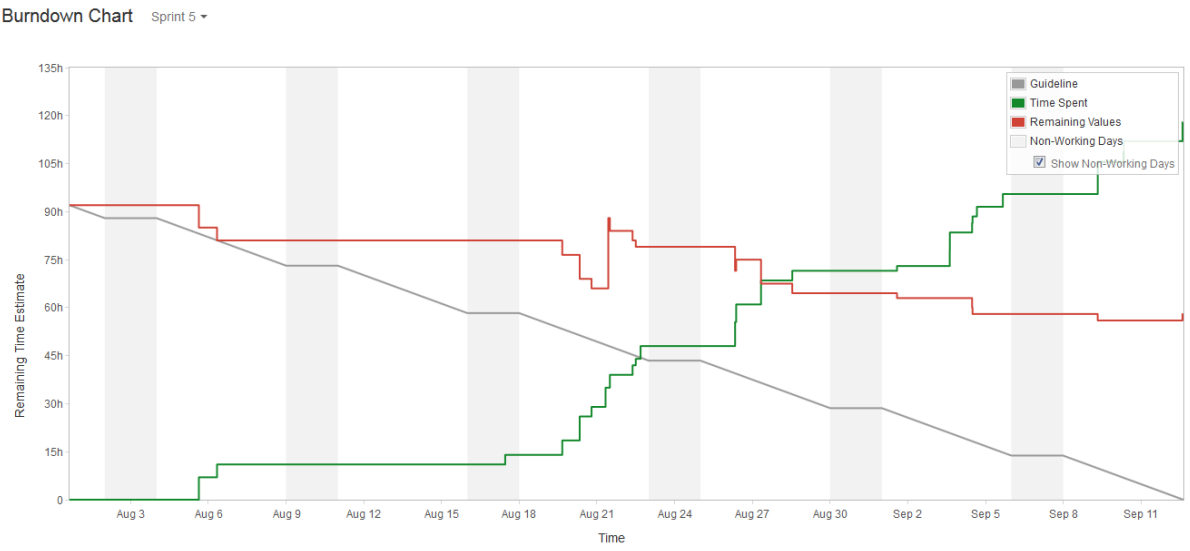
na katerih še nisva delala, seveda v obsegu predvidene hitrosti razvoja. Po tem sva štartala nov sprint.

Med samim sprintom so potekali sestanki Daily Scrum v bolj neformalni izvedbi. Kadar sva bila v prostorih podjetja, sva se ponavadi zjutraj pogovorila o tem, kaj trenutno delava in kje so morebitne težave. V primeru težav sva s pregledom problematike in nasveti poskušala pomagati eden drugemu, kar je velikokrat pripomoglo k hitrejši rešitvi. To je pričalo v dobro dejstvu, da interakcija med člani ekipe "v živo" pripomore k učinkovitejšem reševanju problemov, kar pa zaradi delne porazdeljenosti ekipe na žalost ni bilo vedno mogoče. To težavo je deloma reševala elektronska korespondenca.

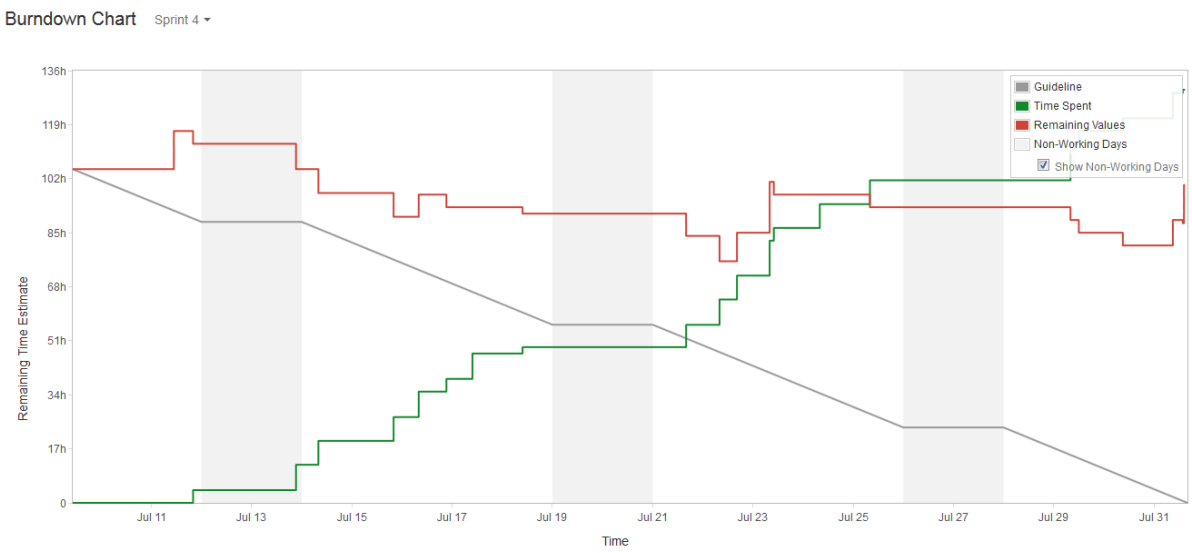
4.2.4 Merjenje napredka

V času iteracije smo na opravilih vnašali opravljeno delo in predvidene ocene dela do zaključka opravila. To je omogočalo spremljanje napredovanja ali nazadovanja hitrosti razvoja (in s tem verjetnost izpolnitve ciljev iteracije) preko grafa Burndown Chart.

Na slikah 4.7. in 4.8. podajam primer enega bolj in drugega manj uspešnih sprintov.



Slika 4.7.: Sprint burndown diagram (Sprint 5)



Slika 4.8.: Sprint burndown diagram (Sprint 4)

Siva črta predstavlja idealno linijo hitrosti razvoja. Rdeča linija predstavlja dejansko vrednost preostalega ur dela za realizacijo, medtem ko zelena linija prikazuje vloženo delo.

V sprintu 5 (Slika 4.7.) smo bili glede na začetku iteracije napovedano hitrost in število dejansko opravljenih točk (ur) še najbolj uspešni, čeprav je rdeča linija še daleč od idealne navkljub temu, da je bilo vloženo več ur dela, kot je bilo napovedano.

V sprintu 4 (Slika 4.8.) pa smo na koncu iteracije končali s praktično enako vrednostjo preostalih točk za realizacijo kot s tisto, s katero smo začeli navkljub temu da je bilo vloženo več dela od napovedanega.

Ti rezultati kažejo na preveč optimistično nastavljena pričakovanja in neupoštevanje zgodovinske vrednosti dejansko realizirane hitrosti razvoja, čemur botrujejo razlogi, ki sem jih že opisal.

4.2.5 Načrtovanje izdaj in roki

Glavni izdaji za podjetje predstavljata različice spletne portala za slovenski trg in različica za tuje trge (v angleščini), ki naj bo pripravljena za lokalizacijo. Začetni rok za slovensko različico glede na ocenjen obseg dela in predvideno hitrost razvoja je bil konec septembra 2014, za tujo različico spletnega portala pa mesec dni kasneje: konec oktobra 2014. Zaradi dejansko manjše realizirane hitrosti sta se roka začela oddaljevati.

Sredi septembra 2014 je bil na željo naročnikov organiziran sestanek, kjer naj bi razvojna ekipa predstavila trenutno stanje projekta in nove napovedi rokov izdaj. Glede na preostala

opravila in maksimalno možno hitrost (kjer ponovno nismo upoštevali zgodovinske vrednosti dejanske hitrosti) so bili predstavljeni novi roki: slovenska različica do konca januarja 2015 in tuja različica do konca februarja 2015. Novo predstavljeni roki naročnikom (predvsem oddelku prodaje) niso ustrezali predvsem zaradi načrtov plasiranja podjetja na tujih trgih,- pri čemer je za lokalizacijo pripravljen spletni portal ključnega pomena.

Zaradi teh razlogov se je začelo razmišljati o opustitvi določenih modulov ali pa povečanju resursov razvoja. Prevladala je odločitev o povečanju resursov in tako smo z manjšo reorganizacijo v podjetju v razvojno ekipo dobili še enega razvijalca, ki je pred tem nastopal v vlogi lastnika produkta (in avtor prejšnje verzije portala).

Naročniki so pri tem predstavili nove željene roke izdaj. Slovenska verzija do konca leta 2014 in tuja verzija do konca januarja 2015. Za izračun rokov izdaj sem pripravil razpredelnico v Excelu, ki jo prikazuje tabela 4.2.

PROJEKT: SPLETNI PORTAL Časovne ocene po modulih			
	Točke (programer/dni)		**Velocity (tedenski doprinos)**
Spletna trgovina	12		Rok 4,5
Implementacija predstavitve izdelkov	23		Teja 3
Uporaba izdelkov	23,5		Boštjan 4,5
Korporativni blog	8		
Responsive design	5		**Skupaj 12**
Varnost spletnega portala	14		
SEO (brez razvojne ekipe)	0		
Povezava na družbena omrežja	3,5		
Vnos vsebin spletnega portala	2,5		
Integracija Magento <-> Vasco	6		**Tednov za realizacijo:**
Skupaj SLO portal	**97,5**		**8,12**
Lokalizacija spletnega portala	9,5		
Prireditev spletnega portala za tujino	11,5		
Izdelava dokumentacije za predajo	4		
EU portal in tujina skupaj	**25**		**2,08**
Poročila Magento	3		
Dodatne plačilne možnosti	16		
Ostalo	**19**		**1,58**

Tabela 4.2.: Izačun potrebnega časa do izdaj

Iz tabele 4.2. je razvidno, koliko časa v tednih je potrebno še do zaključka posameznih izdaj ob predpostavki, da bo skupna realizirana hitrost razvojne ekipe 12 točk na teden. V času

nastanka tega diplomskega dela nam ta izračun še zagotavlja ujemanje z roki, ki so jih postavili notranji naročniki, seveda pa pomeni, da bodo morale biti nadaljnje iteracije realizirane v celoti.

4.2.6 Sodelovanje z lastnikom izdelka in ostalimi udeleženci

Po začetni opredelitvi koncepta prenove spletnega portala in zagotovitvi potrebnih materialov s strani oblikovalke, se lastnika izdelka zaradi drugih obveznosti v podjetju nista mogla redno udejstvovati pri izvedbi projekta, vendar pa sodelovanje ni bilo prekinjeno v celoti. Ko se je iz strani razvojne ekipe nabralo večje število vprašanj oziroma dilem, ki so vplivale na razvoj, smo organizirali skupne sestanke z lastniki in prediskutirali izbrana vprašanja in dileme. Če je šlo tudi za vprašanja glede oblike spletnih strani, sta lastnika na sestanke povabila tudi oblikovalko. Tako smo našli skupne rešitve, ki so ustrezale vsem udeleženim.

Eden od lastnikov izdelka je kot glavni razvijalec stare spletnega portala (in sedaj ponovno član razvojne ekipe) pogosto sodeloval tudi s tehničnimi nasveti razvojni ekipi.

4.2.7 Planirani in dejanski rezultati

Kot smo opazili v prejšnjih podpoglavjih, so se dejanski rezultati na projektu razlikovali od planiranih. Glavne razloge za to vidim v:

- neupoštevanju zgodovinske vrednosti razvojne hitrosti iz strani članov razvojne ekipe pri planiranju iteracij na eni strani in prevelika pričakovanja iz strani naročnikov na drugi strani,
- delni porazdeljenosti ekipe, s katero je nastopilo pomanjkanje komunikacije "v živo" in nedosledno izvajanje dnevnih sestankov,
- nezmožnosti lastnikov produkta, da bi se bolj aktivno udejstvovali na projektu, tako da je del teh aktivnosti morala prevzeti razvojna ekipa,
- motnjah razvojne ekipe, ki nastopijo v delovnem okolju podjetja (kot npr. neplanirani vzdrževalni posegi na obstoječem spletnem portalu, računalniška podpora sodelavcem, etc.).

4.2.8 Pridobivanje izkušenj

Tekom projekta smo pridobili marsikatero izkušnjo pri implementaciji funkcionalnosti za platformo Magento pa tudi nekaj izkušenj pri ocenjevanju uporabniških zgodb in načrtovanju

iteracij. Upam, da bomo te izkušnje v bodoče dobro izkoristili: planirali bolj realno in tudi povečali hitrost razvoja. Samo na ta način bomo lahko dosegli zastavljene roke pričakovanih izdaj.

Poglavje 5 Sklepne ugotovitve

V pričujoči diplomski nalogi sem opisal izvor agilnih metod in kako v zadnjih letih postajajo vse bolj uveljavljene pri vodenju sodobnih projektov razvoja programske opreme. To ne pomeni, da bodo povsem izrinile tradicionalne metode, kaže pa na to, da so zahteve pri razvoju programske opreme pogosto nejasne in se med razvojem spreminjajo, agilne metode pa nam nudijo postopke, s katerimi se lažje spoprimemo s temi izzivi in še vedno pridemo do željenih rezultatov. Eden od razlogov, za katerega menim, da je pomemben pri uspehu agilnih metod, je njihova naravnanost, da programsko opremo približajo ljudem in spodbujajo interakcijo med njimi: tako med člani razvojnih ekip kot tudi med uporabniki.

Podrobneje sem opisal metodo Scrum, ki sodeč po številnih raziskavah postaja najbolj razširjena metoda med uporabniki agilnih metod. Za to je prav gotovo zaslužna njena jasna opredelitev in razumljivost, ki pripomore k enostavni aplikaciji metode na različne projekte. Prav tako pa smo spoznali, kar govori že njena definicija: da je namreč lahka za razumevanje in težka za obvladovanje.

V predstavitvi projekta sem bralca seznanil s poslanstvom in funkcijo novega spletnega portala in kakšni so bili razlogi, ki so podjetje pripeljaji do odločitve o prenovi obstoječe spletnega portala. Nadaljeval sem z bolj tehničnimi vidiki projekta in pri tem podrobneje opisal nekatera sodobna orodja, ki se uporabljajo pri razvoju spletnih pa tudi ostalih aplikacij in rešujejo tudi nekatere probleme (deloma) porazdeljenih ekip.

Posebno podpoglavje je bilo namenjeno popularni platformi za elektronsko poslovanje Magento, njeni arhitekuri in funkciji pri razvoju novega spletnega portala. Na tem mestu bi rad omenil, da smo se za Magento odločili tudi iz časovnih razlogov, saj bi razvoj popolnoma samostojne in lastne platforme od samega začetka po vsej verjetnosi terjal še več časa. Za prihodnost pa še vedno obstaja želja po implementaciji lastne rešitve za elektronsko poslovanje, kar bi nudilo večji nadzor nad funkcionalnostmi in večjo prilagodljivost podjetju specifičnim poslovnim procesom. Magento sicer lahko ponudi hitro rešitev za vzpostavitev elektronskega poslovanja, sploh, če se dogradi z nekaterimi razširitvami, ki so na voljo na svetovnem spletu. Čim pa se lotimo lastnih prilagoditev, lahko hitro postanemo "žrtve" njegove kompleksne arhitekture.

V četrtem poglavju sem opisal za to diplomsko nalogo bistveni del projekta: uvajanje metode Scrum na projekt prenove spletnega portala: od tega, kako se je formirala ekipa in razdelile vloge, preko izbora aplikacije za vodenje projekta do tega, kako je projekt potekal in kako smo merili napredek tekom sedmih dosedanjih iteracij (sprintov).

Motivacija za uvedbo Scruma v projekt je bila močna, saj se do začetka tega projekta v podjetju pri vodenju razvoja programskih rešitev še niso uporabljale metode za vodenje projektov. Ker je bilo to večinoma delo posameznikov, to verjetno niti ni bilo tako potrebno. Če pa smo z novo formirano ekipo hoteli priti do nekih merljivih rezulatov in izračunati vsaj približne roke dostav, je bilo to nujno potrebno. Sama narava projekta pa je v skladu s sodobnimi metodami govorila v prid izbiri Scruma.

Ker pa je šlo za prvi tak projekt in okoliščine niso dopuščale popolne implementacije metode Scrum, ni vse teklo po načrtu. Spoznali smo, da je se priporočil metode dobro držati in redno spremljati stanje projekta, da bi lahko hitreje odreagirali. Kot govori eno od priporočil za velikost razvojne ekipe od 3 do 9 članov, lahko potrdim, da je v ekipi 2 članov interakcija nekoliko manjša in se s prihodom tretjega člana že pozna povečan obseg interakcije in s tem hirejša pot do rešitev. Sem mnenja, da interakcijo zmanjšuje tudi delna porazdeljenost ekipe, vendar trenutna situacija v podjetju še ne kaže v prid združitve ekipe za več dni v tednu na enem mestu. Prav tako bi nam koristilo, če bi lahko lastnik produkta odigral bolj aktivno vlogo.

V bodoče upam, da bomo člani ekipe metodo Scrum še bolje ponotranjili in da se bo uveljavila kot stalna praksa, s pomočjo katere bomo tekom iteracij izboljševali proces razvoja, napovedovali točne roke in dostavili kvalitetno programsko opremo. Na ta način bomo izpopolnjevali tudi sebe in ponudili uporabniku prijazno izkušnjo.

Literatura

[1] Rok Alidžanović, Projekt implementacije novega spletnega portala: Product Backlog (začetna zbirka uporabniških zgodb), Ljubljana: notranja dokumentacija podjetja, 2014.

[2] David Anderson, *Declaration of Interdependence*, 2005. Declaration of Interdependence [Online]. Dosegljivo: http://pmdoi.org/.

[3] Mike Cohn, *User Stories Applied: For Agile Software Development*. Addison Wesley, mar. 2004.

[4] Nurul Ferdous, *Magento 1.4 Development Cookbook*, Birmingham: Packt Publishing, dec. 2010.

[5] Peter Kuralt, *Predlog konceptualne opredelitve prenovljene spletnega portala*, Ljubljana: notranja dokumentacija podjetja, 2013.

[6] Viljan Mahnič, *Prosojnice s predavanj pri predmetu Tehnologija programske opreme*, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, študijsko leto 2013/2014.

[7] V. Mahnič, N. Žabkar, *Measuring Progress of Scrum-bases Software Projects*. Elektronika ir Elektrotechnika, Vol. 18, No. 8, 2012.

[8] T. E. Murphy, J. Duggan, D. Norton, B. Prentice, D. C. Plummer, S. Landry, "Predicts 2010: Agile and Cloud Impact Application Development Directions", Gartner, Dec., 2009.

[9] Jeff Sutherland and Ken Schwaber, *The Scrum Guide*. Scrum Guides [Online]. Dosegljivo: http://www.scrumguides.org/.

[10] D. West, T. Grant, "Agile Development: Mainstream adoption has changed agility, Trends in real-world adoption of agile methods", *Forrester*, Jan. 20, 2010, [Online]. Dosegljivo:

http://www.forrester.com/rb/Research/agile_development_mainstream_adoption_has_changed_agility/q/id/56100/t/2/.

[11] *Agilefant [Online]*. Dosegljivo: http://agilefant.com/.

[12] Agile software development. *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Agile_software_development.

[13] Agilo For Scrum. *Agilo Software [Online]*. Dosegljivo: http://agilosoftware.com/

[14] Capability Maturity Model. *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Capability_Maturity_Model

[15] Centralized Workflow. *Atlassian Git Tutorial [Online]*. Dosegljivo: https://www.atlassian.com/git/tutorials/comparing-workflows/centralized-workflow

[16] Chaos Report. *VersionOne [Online]* Dosegljivo: http://www.versionone.com/assets/img/files/ChaosManifest_2011.pdf

[17] Feature-driven development. *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Feature-driven_development/.

[18] Git (software). *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Git_%28software%29

[19] ISO 9001 Sistemi vodenja kakovosti. *SIQ [Online]*. Dosegljivo: http://www.siq.si/ocenjevanje_sistemov_vodenja/storitve/sistemi_vodenja_kakovosti/.

[20] JIRA. *Atlassian [Online]*. Dosegljivo: https://www.atlassian.com/software/jira/.

[21] JIRA. *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/JIRA

[22] Lean Software Development. *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Lean_software_development/.

[23] Magento. *Wikipedia* [Online]. Dosegljivo: http://en.wikipedia.org/wiki/Magento.

[24] Magento Architecture. *Magento Commerce* [Online]. Dosegljivo: http://www.magentocommerce.com/wiki/welcome_to_the_magento_user_s_guide/chapter_1#magento_s_architecture

[25] Magento Extension Developer's Guide, X.commerce, Inc, 2012 [Online]. Dosegljivo: http://info.magento.com/rs/magentocommerce/images/Magento-Extension-Developers-Guide-v1.0.pdf

[26] Magento is now powering 1 percent of all websites. *W3Techs* [Online]. Dosegljivo: http://w3techs.com/blog/entry/magento_is_now_powering_1_percent_of_all_websites

[27] Manifesto for Agile Software Development. *Agile Manifesto [Online]*. Dosegljivo: http://agilemanifesto.org/.

[28] Manifesto for Software Craftsmanship. *Manifesto for Software Craftsmanship [Online]*. Dosegljivo: http://manifesto.softwarecraftsmanship.org/.

[29] Mingle. Agile Project Management. *Thoughtworks [Online]*. Dosegljivo: http://www.thoughtworks.com/products/mingle-agile-project-management/.

[30] *PHP [Online]*. Dosegljivo: http://php.net/.

[31] Planning and Estimating work for an Agile team. JIRA dokumentacija [Online]. Dosegljivo: https://confluence.atlassian.com/display/AGILE/Tutorial+-+Planning+and+Estimating+work+for+an+Agile+team

[32] Rally. *Rally Software [Online]*. Dosegljivo: https://www.rallydev.com/.

[33] *Scrum.org [Online]*. Dosegljivo: https://www.scrum.org/.

[34] Scrum (software development). *Wikipedia [Online]*. Dosegljivo: http://en.wikipedia.org/wiki/Scrum_%28software_development%29/.

[35] *State Of Agile*. *Why Agile? [Online]*. Dosegljivo: http://stateofagile.versionone.com/why-agile/.

[36] State of Agile. *Version One [Online]*. Dosegljivo: http://stateofagile.versionone.com/agile-practices-tools/

[37] Usage statistics and market share of Magento for websites. *W3Techs* [Online]. Dosegljivo: http://w3techs.com/technologies/details/cm-magento/all/all.

[38] *Zend Framework [Online]*. Dosegljivo: http://framework.zend.com/.